

DDDDDDDDDDDDDD	IIIIIIIIII	FFFFFFFFFFFFFFFF
DDDDDDDDDDDDDD	IIIIIIIIII	FFFFFFFFFFFFFFFF
DDDDDDDDDDDDDD	IIIIIIIIII	FFFFFFFFFFFFFFFF
DDD	III	FFF
DDD	III	FFF
DDD	III	FFF
DDD	III	FFF
DDD	III	FFF
DDD	III	FFF
DDD	III	FFF
DDD	III	FFFFFFFFFFFFFF
DDD	III	FFFFFFFFFFFFFF
DDD	III	FFFFFFFFFFFFFF
DDD	III	FFF
DDD	III	FFF
DDD	III	FFF
DDD	III	FFF
DDD	III	FFF
DDD	III	FFF
DDD	III	FFF
DDDDDDDDDDDDDD	IIIIIIIIII	FFF
DDDDDDDDDDDDDD	IIIIIIIIII	FFF
DDDDDDDDDDDDDD	IIIIIIIIII	FFF

DIF
VO4

[illegible][illegible]

```
0001 0 MODULE DIF_OUTPUT ( ! Differences output routines
0002 0     LANGUAGE (BLISS32),
0003 0     ADDRESSING_MODE (EXTERNAL=GENERAL,
0004 0     NONEXTERNAL=LONG_RELATIVE),
0005 0     IDENT = 'V04-000'
0006 0 ) =
0007 1 BEGIN
0008 1 .....
0009 1 *
0010 1 * COPYRIGHT (c) 1978, 1980, 1982, 1984 BY
0011 1 * DIGITAL EQUIPMENT CORPORATION, MAYNARD, MASSACHUSETTS.
0012 1 * ALL RIGHTS RESERVED.
0013 1 *
0014 1 * THIS SOFTWARE IS FURNISHED UNDER A LICENSE AND MAY BE USED AND COPIED
0015 1 * ONLY IN ACCORDANCE WITH THE TERMS OF SUCH LICENSE AND WITH THE
0016 1 * INCLUSION OF THE ABOVE COPYRIGHT NOTICE. THIS SOFTWARE OR ANY OTHER
0017 1 * COPIES THEREOF MAY NOT BE PROVIDED OR OTHERWISE MADE AVAILABLE TO ANY
0018 1 * OTHER PERSON. NO TITLE TO AND OWNERSHIP OF THE SOFTWARE IS HEREBY
0019 1 * TRANSFERRED.
0020 1 *
0021 1 * THE INFORMATION IN THIS SOFTWARE IS SUBJECT TO CHANGE WITHOUT NOTICE
0022 1 * AND SHOULD NOT BE CONSTRUED AS A COMMITMENT BY DIGITAL EQUIPMENT
0023 1 * CORPORATION.
0024 1 *
0025 1 * DIGITAL ASSUMES NO RESPONSIBILITY FOR THE USE OR RELIABILITY OF ITS
0026 1 * SOFTWARE ON EQUIPMENT WHICH IS NOT SUPPLIED BY DIGITAL.
0027 1 *
0028 1 *
0029 1 .....
0030 1
0031 1 ++
0032 1
0033 1 FACILITY:      DCL Differences command
0034 1
0035 1 ABSTRACT:
0036 1     The DCL DIFFERENCES command compares the contents of
0037 1     two files.
0038 1
0039 1 ENVIRONMENT:
0040 1     VAX native, user mode
0041 1
0042 1 --
0043 1
0044 1 AUTHOR:      Peter George, Benn Schreiber      CREATION DATE: 1-August-1981
0045 1
0046 1 MODIFIED BY:
0047 1
0048 1     V03-002 PCG0002      Peter George      11-Apr-1984
0049 1     Fix ACCVIO from DIFF/MERGED=0.
0050 1
0051 1     V03-001 PCG0001      Peter George      13-Oct-1982
0052 1     Skip matches that precede a merged output section.
0053 1     Do not put change bars in front of ignored headers.
0054 1     Never output ignored records.
0055 1
0056 1 --
```



```
58 0057 1 LIBRARY
59 0058 1 'SYSSLIBRARY:STARLET.L32';
60 0059 1
61 0060 1 REQUIRE
62 0061 1 'DIFPRE'; ! DIF prefix file
63 0135 1 REQUIRE 'DIFDEF'; ! DIF data structures
64 0136 1
65 0367 1
66 0368 1
67 0369 1 ! Difference global data
68 0370 1
69 0371 1 EXTERNAL
70 0372 1     dif$gl_commdesc : BBLOCK, ! Desc for buffer of comment delimiters
71 0373 1     dif$gl_cmddesc : BBLOCK, ! Descriptor for command line
72 0374 1     dif$gl_ignore : BBLOCK, ! Flags of characters to ignore
73 0375 1     dif$gl_header, ! No. of lines to skip as header
74 0376 1     dif$gl_match, ! No. of records that constitute a match
75 0377 1     dif$gl_maxdif, ! Maximum number of unmatched records
76 0378 1     dif$gl_merged, ! No. of matched lines to follow each list of differences
77 0379 1     dif$gl_parallel, ! Same as above for parallel
78 0380 1     dif$gl_wndwsiz, ! No. of records to search before
79 0381 1     ! declaring a mismatch
80 0382 1     dif$gl_flags : BBLOCK, ! Flags
81 0383 1     dif$gl_difrec, ! No. of difference records found
82 0384 1     dif$gl_difsec, ! No. of difference sections detected
83 0385 1     dif$gl_dumpwidth, ! Width of hex/octal data part of line
84 0386 1     dif$gl_entsperline, ! No. of entries on a hex/octal line
85 0387 1     dif$gl_width, ! Width of lines in output listing
86 0388 1     dif$gl_parwidth, ! Width of lines in parallel listing
87 0389 1     dif$gl_inbuf, ! Address of the input record buffer
88 0390 1     dif$gl_outbuf : REF VECTOR [, BYTE], ! Address of the output record buffer
89 0391 1     dif$gl_faofullbuf : BBLOCK, ! Hex/octal fao full line buffer
90 0392 1     dif$gl_faopartbuf : BBLOCK, ! Hex/octal fao partial line buffer
91 0393 1     dif$gl_faofulldesc : BBLOCK, ! String desc for hex/octal full output line
92 0394 1     dif$gl_faopartdesc : BBLOCK, ! String desc for hex/octal partial output line
93 0395 1
94 0396 1 ! Input and output file data structures
95 0397 1
96 0398 1     dif$gl_masdesc : BBLOCK, ! String desc for input file
97 0399 1     dif$gl_masfdb : BBLOCK, ! Master file fdb
98 0400 1     dif$gl_masrab : BBLOCK, ! RAB for master file
99 0401 1     dif$gl_revdesc : BBLOCK, ! String desc for revision file
100 0402 1     dif$gl_revfdb : BBLOCK, ! Revision file fdb
101 0403 1     dif$gl_revrab : BBLOCK, ! RAB for revision file
102 0404 1     dif$gl_outdesc : BBLOCK, ! String desc for output file
103 0405 1     dif$gl_outrab : BBLOCK; ! RAB for output file
104 0406 1
105 0407 1 EXTERNAL ROUTINE
106 0408 1     dif$format_hex_octal, ! Format a record in either hex or octal
107 0409 1     ots$cvl_l_i, ! Convert longword to text
108 0410 1     sys$fao; ! FAO conversion routine
109 0411 1
110 0412 1 FORWARD ROUTINE
111 0413 1     additional_output, ! Output 2nd, 3rd, ... listings
112 0414 1     get_rfa_text, ! Get record text using RFA
113 0415 1     init_hex_octal, ! Prepare for hex or octal output
114 0416 1     insert_lineno, ! Insert a line # in output line
```

```
115 0417 1 output_changebar,      ! Output in change bar format
116 0418 1 output_listing_trailer, ! Output listing trailer
117 0419 1 output_cmdfile,         ! Output file line of trailer
118 0420 1 output_cmdfao,          ! Output command using SFAO
119 0421 1 output_cmdcounted,      ! Output counted entity
120 0422 1 output_cmdentity,       ! Output command line entity
121 0423 1 output_merged,          ! Output in merged format
122 0424 1 output_parallel,        ! Output in parallel format
123 0425 1 output_separated,       ! Output in separated format
124 0426 1 output_slp,             ! Output in SLP format
125 0427 1 put_blank,              ! Output a blank line
126 0428 1 put_desc,               ! Output a descriptor
127 0429 1 put_hex_octal_header,   ! Output hex/octal record header
128 0430 1 put_idline,             ! Output a file id line
129 0431 1 put_parallel_idline,    ! Output a parallel file id line
130 0432 1 put_record,             ! Output a record in appropriate radix
131 0433 1 put_record_ascii,       ! Output an ascii record
132 0434 1 put_record_hex_octal,   ! Output a hex or octal record
133 0435 1 put_record_parallel,    ! Output a parallel format record
134 0436 1 translate_tabs,         ! Convert tabs to blanks
135 0437 1 write_mismatch;         ! Output records in a mismatch
136 0438 1
137 0439 1 OWN
138 0440 1 cmd_bufpos,              ! Position for command output
139 0441 1 cr : COUNTEDSTRING ('<CR>'),
140 0442 1 ff : COUNTEDSTRING ('<FF>'),
141 0443 1 lf : COUNTEDSTRING ('<LF>'),
142 0444 1 vt : COUNTEDSTRING ('<VT>'),
143 0445 1 blanks : COUNTEDSTRING (' '),
144 0446 1 period : COUNTEDSTRING ('.'),
145 0447 1 change : COUNTEDSTRING ('***CHANGE***'),
146 0448 1 difrec : COUNTEDSTRING ('Number of difference records found: !ZL'),
147 0449 1 difsec : COUNTEDSTRING ('Number of difference sections found: !ZL'),
148 0450 1 file : COUNTEDSTRING ('File '),
149 0451 1 hexfull : COUNTEDSTRING ('!!!ZL(9XL) !!!ZLAF !!!6XL'),
150 0452 1 hexheader : COUNTEDSTRING ('RECORD NUMBER !ZL (!8XL) LENGTH !ZL (!8XL) !AS'),
151 0453 1 hexpart : COUNTEDSTRING ('!!!!ZL(9XL) !!!ZLAF !!!6XL'),
152 0454 1 octfull : COUNTEDSTRING ('!!!ZL(12OL) !!!ZLAF !!!6OL'),
153 0455 1 octheader : COUNTEDSTRING ('RECORD NUMBER !ZL (!8OL) LENGTH !ZL (!8OL) !AS'),
154 0456 1 octpart : COUNTEDSTRING ('!!!!ZL(12OL) !!!ZLAF !!!6OL'),
155 0457 1 slpoprs : COUNTEDSTRING ('-\\%>/<'),
156 0458 1 stars : COUNTEDSTRING ('*****');
157 0459 1
158 0460 1 LITERAL
159 0461 1 ffeed = 12;              ! form feed character
160 0462 1
161 0463 1 EXTERNAL LITERAL
162 0464 1 dif$_readerr,
163 0465 1 dif$_writeerr;
164 0466 1
```



```
166 0467 1 GLOBAL ROUTINE write_mismatch =
167 0468 2 BEGIN
168 0469
169 0470 1++
170 0471
171 0472 FUNCTIONAL DESCRIPTION:
172 0473
173 0474 When the end of a mismatch is detected, this routine is used to
174 0475 call the output routine for the appropriate user-specified output
175 0476 format. The order of the IF - THEN - ELSE statements is significant
176 0477 because this control structure is imbedded in several other places
177 0478 in the Diff code.
178 0479
179 0480 INPUTS:
180 0481
181 0482 None
182 0483
183 0484 OUTPUTS:
184 0485
185 0486 dif$gl_difsec is incremented
186 0487
187 0488 ROUTINE VALUES:
188 0489
189 0490 Always true
190 0491
191 0492 --
192 0493 dif$gl_difsec = .dif$gl_difsec + 1; ! Incr count of diff sections
193 0494
194 0495 IF .dif$gl_flags [dif$v_parallel] ! Call appropriate routine
195 0496 THEN output_parallel ()
196 0497
197 0498 ELSE IF .dif$gl_flags [dif$v_merged]
198 0499 THEN output_merged ()
199 0500
200 0501 ELSE IF .dif$gl_masfdb [fdb$v_separated]
201 0502 THEN output_separated (dif$gl_masfdb)
202 0503
203 0504 ELSE IF .dif$gl_revfdb [fdb$v_separated]
204 0505 THEN output_separated (dif$gl_revfdb)
205 0506
206 0507 ELSE IF .dif$gl_masfdb [fdb$v_changebar]
207 0508 THEN output_changebar (dif$gl_masfdb)
208 0509
209 0510 ELSE IF .dif$gl_revfdb [fdb$v_changebar]
210 0511 THEN output_changebar (dif$gl_revfdb)
211 0512
212 0513 ELSE IF .dif$gl_flags [dif$v_slp]
213 0514 THEN output_slp ();
214 0515
215 0516 RETURN true;
216 0517 1 END;
```

```
.TITLE DIF_OUTPUT
.IDENT \V04-000\
.PSECT $OWNS,NOEXE,2
```

```
00000 CMD_BUFPOS:
                                .BLKB 4
                                .BYTE 4
3E 52 43 04 00004 CR: .ASCII \<CR>\
00005 .BLKB 3
00009 .BYTE 4
3E 46 46 04 0000C FF: .ASCII \<FF>\
0000D .BLKB 3
00011 .BYTE 4
3E 46 4C 04 00014 LF: .ASCII \<LF>\
00015 .BLKB 3
00019 .BYTE 4
3E 54 56 04 0001C VT: .ASCII \<VT>\
0001D .BLKB 3
00021 .BYTE 8
20 20 20 20 20 20 20 08 00024 BLANKS: .ASCII \
00025 .BLKB 3
0002D .BYTE 1
01 00030 PERIOD: .ASCII \.
00031 .BLKB 2
00032 .BYTE 13
0D 00034 CHANGE: .ASCII \ ***CHANGE***\
00035 .BLKB 2
00042 .BYTE 39
65 66 66 69 64 20 66 6F 20 72 65 62 6D 75 4E 00044 DIFREC: .ASCII \Number of difference records found: !ZL\
66 20 73 64 72 6F 63 65 72 20 65 63 6E 65 72 00045 .BLKB 2
66 20 73 6E 6F 69 74 63 65 73 20 65 63 6E 65 72 00054 .BYTE 39
20 73 6E 6F 69 74 63 65 73 20 65 63 6E 65 72 00063 .ASCII \Number of difference sections found: !ZL\
20 73 6E 6F 69 74 63 65 73 20 65 63 6E 65 72 0006C .BLKB 3
20 73 6E 6F 69 74 63 65 73 20 65 63 6E 65 72 0006D .BYTE 5
05 00098 FILE: .ASCII \File \
20 65 6C 69 46 00099 .BLKB 2
0009E .BYTE 24
5A 21 21 21 20 29 4C 58 39 28 4C 5A 21 21 21 000A0 HEXFULL: .ASCII \!!!ZL(9XL) !!!ZLAF !!6XL\
21 21 21 21 20 29 4C 58 36 21 21 20 46 41 4C 000A1 .BLKB 3
000B0 .BYTE 46
000B9 .ASCII \RECORD NUMBER !ZL (!8XL) LENGTH !ZL (!8XL)
2E 000BC HEXHEADER: .BLKB 1
000BD .BYTE 31
21 20 52 45 42 4D 55 4E 20 44 52 4F 43 45 52 000CC .ASCII \L) !AS\
54 47 4E 45 4C 20 29 4C 58 38 21 28 20 4C 5A 000DB .BLKB 1
000DB .BYTE 31
21 20 52 45 42 4D 55 4E 20 44 52 4F 43 45 52 000E5 .ASCII \!!!!!!ZL(9XL) !!!ZLAF !!!6XL\
21 20 52 45 42 4D 55 4E 20 44 52 4F 43 45 52 000EB .BLKB 1
21 20 52 45 42 4D 55 4E 20 44 52 4F 43 45 52 000EC HEXPART: .BYTE 25
58 36 21 21 21 21 20 46 41 4C 5A 21 21 21 21 000ED .ASCII \!!!!!!ZL(120L) !!!ZLAF !!60L\
58 36 21 21 21 21 20 46 41 4C 5A 21 21 21 21 000FC .BLKB 2
21 21 21 20 29 4C 4F 32 31 28 4C 5A 21 21 21 21 0010C OCTFULL: .BYTE 46
21 21 21 20 29 4C 4F 36 21 21 20 46 41 4C 5A 0010D .ASCII \
21 21 21 20 29 4C 4F 36 21 21 20 46 41 4C 5A 0011C .BLKB 2
2E 00128 OCTHEADER: .BYTE 46
```



```
21 20 52 45 42 4D 55 4E 20 44 52 4F 43 45 52 00129
54 47 4E 45 4C 20 29 4C 4F 38 21 28 20 4C 5A 00138
      4F 38 21 28 20 4C 5A 21 20 48 00147
      53 41 21 20 29 4C 00151
      00157
      00158 OCTPART:
      00159
      00160
      00177
      00179
      0017C
      0017D
      00183
      0C 00184
      2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 2A 00185
```

```
.ASCII \RECORD NUMBER !ZL (!80L) LENGTH !ZL (!80L) :
      .ASCII \L) !AS\ :
      .BLKB 1 :
      .BYTE 32 :
      .ASCII \!!!!!!ZL(120L) !!!!!ZLAF !!!!!60L\ :
      .BLKB 3 :
      .BYTE 6 :
      .ASCII \- \<92> \x0 /<\ :
      .BLKB 1 :
      .BYTE 12 :
      .ASCII \*****\ :
```

```
.EXTRN DIF$GL_COMMDESC
.EXTRN DIF$GL_CMDESC, DIF$GL_IGNORE
.EXTRN DIF$GL_HEADER, DIF$GL_MATCH
.EXTRN DIF$GL_MAXDIF, DIF$GL_MERGED
.EXTRN DIF$GL_PARALLEL
.EXTRN DIF$GL_WNDSIZ, DIF$GL_FLAGS
.EXTRN DIF$GL_DIFREC, DIF$GL_DIFSEC
.EXTRN DIF$GL_DUMPWIDTH
.EXTRN DIF$GL_ENTSPERLINE
.EXTRN DIF$GL_WIDTH, DIF$GL_PARWIDTH
.EXTRN DIF$GL_INBUF, DIF$GL_OUTBUF
.EXTRN DIF$GL_FAOFULLBUF
.EXTRN DIF$GL_FAOPTARTBUF
.EXTRN DIF$GL_FAOFULLDESC
.EXTRN DIF$GL_FAOPTARTDESC
.EXTRN DIF$GL_MASDESC, DIF$GL_MASFDB
.EXTRN DIF$GL_MASRAB, DIF$GL_REVDESC
.EXTRN DIF$GL_REVFDB, DIF$GL_REVRAB
.EXTRN DIF$GL_OUTDESC, DIF$GL_OUTRAB
.EXTRN DIF$FORMAT_HEX_OCTAL
.EXTRN OTSSCVT_L_T1, SYSSFAO
.EXTRN DIF$_READERR, DIF$_WRITEERR
```

.PSECT \$CODES, NOWRT, 2

```
001C 00000
54 00000000G 00 9E 00002
53 00000000G 00 9E 00009
52 00000000G 00 9E 00010
      00000000G 00 D6 00017
09 00000000V 64 06 E1 0001D
      EF 00 FB 00021
      48 11 00028
09 00000000V 64 05 E1 0002A 1$:
      EF 00 FB 0002E
      3B 11 00035
05 62 02 E1 00037 2$:
      DC A2 9F 0003B
      07 11 0003E
0C 63 02 E1 00040 3$:
      DC A3 9F 00044
```

```
.ENTRY WRITE_MISMATCH, Save R2,R3,R4 : 0467
MOVAB DIF$GL_FLAGS, R4 :
MOVAB DIF$GL_REVFDB+36, R3 :
MOVAB DIF$GL_MASFDB+36, R2 :
INCL DIF$GL_DIFSEC : 0493
BBC #6, DIF$GL_FLAGS, 1$ : 0495
CALLS #0, OUTPUT_PARALLEL : 0496
BRB 9$ :
BBC #5, DIF$GL_FLAGS, 2$ : 0498
CALLS #0, OUTPUT_MERGED : 0499
BRB 9$ :
BBC #2, DIF$GL_MASFDB+36, 3$ : 0501
PUSHAB DIF$GL_MASFDB : 0502
BRB 4$ :
BBC #2, DIF$GL_REVFDB+36, 5$ : 0504
PUSHAB DIF$GL_REVFDB : 0505
```


00000000V	EF		01	FB	00047	4\$:	CALLS	#1, OUTPUT_SEPARATED	:	
			22	11	0004E		BRB	9\$:	
	05		62	E9	00050	5\$:	BLBC	DIF\$GL_MASFDB+36, 6\$:	0507
		DC	A2	9F	00053		PUSHAB	DIF\$GL_MASFDB	:	0508
			06	11	00056		BRB	7\$:	
	0C		63	E9	00058	6\$:	BLBC	DIF\$GL_REVFDB+36, 8\$:	0510
		DC	A3	9F	0005B		PUSHAB	DIF\$GL_REVFDB	:	0511
00000000V	EF		01	FB	0005E	7\$:	CALLS	#1, OUTPUT_CHANGEAB	:	
			0B	11	00065		BRB	9\$:	
	07	01	A4	E9	00067	8\$:	BLBC	DIF\$GL_FLAGS+1, 9\$:	0513
00000000V	EF		00	FB	0006B		CALLS	#0, OUTPUT_SL?	:	0514
	50		01	D0	00072	9\$:	MOVL	#1, R0	:	0516
			04	00075			RET		:	0517

; Routine Size: 118 bytes, Routine Base: \$CODE\$ + 0000

```
218 0518 1 GLOBAL ROUTINE additional_output =
219 0519 BEGIN
220 0520
221 0521 **
222 0522
223 0523 FUNCTIONAL DESCRIPTION:
224 0524
225 0525     Output any additional listings, i.e. additional formats or radices,
226 0526     that have been requested.
227 0527
228 0528 INPUTS:
229 0529
230 0530     None
231 0531
232 0532 OUTPUTS:
233 0533
234 0534     None
235 0535
236 0536 ROUTINE VALUES:
237 0537
238 0538     Always true
239 0539
240 0540 --
241 0541
242 0542 LOCAL
243 0543     first,                                ! Flag set if finishing first listing
244 0544     save_flags : BBLOCK [4],             ! save dif$gl_flags
245 0545     dashdesc : BBLOCK [dsc$_s_bln],      ! Descriptor for dashes
246 0546     stardesc : BBLOCK [dsc$_s_bln];       ! Descriptor for stars
247 0547
248 0548
249 0549     If SLP output, then cannot have any more output forms, so return.
250 0550
251 0551     IF .dif$gl_flags [dif$_slp]
252 0552     THEN RETURN true;
253 0553
254 0554
255 0555     Set up descriptor for stars.
256 0556
257 0557     stardesc [dsc$_length] = .stars [0];
258 0558     stardesc [dsc$_pointer] = stars [1];
259 0559
260 0560
261 0561     If init flag is still set, then no differences were found.
262 0562     So, cleanup current listing, but don't output any additional listings.
263 0563
264 0564     IF .dif$gl_flags [dif$_init]
265 0565     THEN BEGIN
266 0566         IF NOT .dif$gl_flags [dif$_parallel]
267 0567         AND NOT .dif$gl_flags [dif$_merged]
268 0568         AND NOT .dif$gl_masfdb [fdb$_separated]
269 0569         AND NOT .dif$gl_revfdb [fdb$_separated]
270 0570         THEN BEGIN
271 0571             put_desc (stardesc);
272 0572             put_blank ();
273 0573             END;
274 0574     RETURN true;
```

```
275 0575      END;
276 0576
277 0577
278 0578      If parallel output, then do special processing.
279 0579
280 0580      IF .dif$gl_flags [dif$u_parallel]
281 0581      THEN BEGIN
282 0582          first = false;                                ! Note that first listing is now done
283 0583          dashdesc [dsc$w_length] = .dif$gl_parwidth;    ! Build and output line of dashes
284 0584          dashdesc [dsc$a_pointer] = .dif$gl_outbuf;
285 0585          CH$FILL (ASCII "-", .dif$gl_parwidth, .dif$gl_outbuf);
286 0586          put_desc (dashdesc);
287 0587          put_blank ();                                ! Output a blank line
288 0588          IF (NOT .dif$gl_masfdb [fdb$u_move] AND NOT .dif$gl_revfdb [fdb$u_move])
289 0589              THEN RETURN true;                        ! NO more listings? - then return
290 0590      END
291 0591      ELSE first = true;                                ! Note that first listing is not yet finished
292 0592
293 0593
294 0594
295 0595      If no more output, then use common (non-PARALLEL, non-SLP) ending.
296 0596      and return.
297 0597
298 0598      IF (NOT .dif$gl_masfdb [fdb$u_move] AND NOT .dif$gl_revfdb [fdb$u_move])
299 0599      THEN BEGIN
300 0600          IF NOT .dif$gl_flags [dif$u_merged]            ! Output stars for all but MERGED
301 0601              THEN put_desc (star_desc);
302 0602          put_blank ();                                ! Output a blank line
303 0603          RETURN true;                                  ! All done
304 0604      END;
305 0605
306 0606
307 0607      Save flags so they can be restored later
308 0608
309 0609      save_flags = .dif$gl_flags;
310 0610
311 0611
312 0612      Loop for all possible output formats.
313 0613
314 0614      INCR i FROM 1 TO 3                                ! For each radix
315 0615      DO BEGIN
316 0616
317 0617          IF ((.i EQL 1) AND .dif$gl_flags [dif$u_ascii]) OR    ! If radix specified
318 0618              ((.i EQL 2) AND .dif$gl_flags [dif$u_hex]) OR
319 0619              ((.i EQL 3) AND .dif$gl_flags [dif$u_octal])
320 0620      THEN BEGIN                                            ! Then generate requested listings
321 0621
322 0622          IF (.i NEQ 1) AND (NOT .first)                    ! If hex or octal, and not already init'd
323 0623              THEN init_hex_octal ();                      ! Then init
324 0624
325 0625
326 0626      For each listing format,
327 0627
328 0628          1. If first listing, then
329 0629              reset flag,
330 0630              output terminating lines.
331 0631
```


2. If not first listing, then
 set init flag so that header will be output
 output listing,
 output terminating lines.

```
IF .dif$gl_flags [dif$u_merged]          ! Generate MERGED output
THEN BEGIN
  IF .first
  THEN first = false
  ELSE BEGIN
    dif$gl_flags [dif$u_init] = true;
    output_merged ();
  END;
  put_blank ();
END;

IF .dif$gl_masfdb [fdb$u_separated]        ! Generate SEPARATED output for master file
THEN BEGIN
  IF .first
  THEN first = false
  ELSE BEGIN
    dif$gl_flags [dif$u_init] = true;
    output_separated (dif$gl_masfdb);
  END;
  put_desc (stardesc);
  put_blank ();
END;

IF .dif$gl_revfdb [fdb$u_separated]        ! Generate SEPARATED output for revision file
THEN BEGIN
  IF .first
  THEN first = false
  ELSE BEGIN
    dif$gl_flags [dif$u_init] = true;
    output_separated (dif$gl_revfdb);
  END;
  put_desc (stardesc);
  put_blank ();
END;

IF .dif$gl_masfdb [fdb$u_changebar]        ! Generate CHANGE_BAR output for master file
THEN BEGIN
  IF .first
  THEN first = false
  ELSE BEGIN
    dif$gl_flags [dif$u_init] = true;
    output_changebar (dif$gl_masfdb);
  END;
  put_desc (stardesc);
  put_blank ();
END;

IF .dif$gl_revfdb [fdb$u_changebar]        ! Generate CHANGE_BAR output for revision file
THEN BEGIN
  IF .first
  THEN first = false
```

```
0689 ELSE BEGIN
0690     dif$gl_flags [dif$u_init] = true;
0691     output_changebar (dif$gl_revfdb);
0692     END;
0693     put_desc (stardesc);
0694     put_blank ();
0695     END;
0696
0697 END;
0698
0699
0700 If done outputting a radix, then clear that bit.
0701
0702 IF 1 EQL 1
0703 THEN dif$gl_flags [dif$u_ascii] = false
0704 ELSE IF 1 EQL 2
0705 THEN dif$gl_flags [dif$u_hex] = false
0706 ELSE dif$gl_flags [dif$u_octal] = false;
0707
0708 END;
0709
0710
0711 restore flags
0712
0713 dif$gl_flags [dif$u_ascii] = .save_flags [dif$u_ascii];
0714 dif$gl_flags [dif$u_hex] = .save_flags [dif$u_hex];
0715 dif$gl_flags [dif$u_octal] = .save_flags [dif$u_octal];
0716
0717 RETURN true;
0718 END;
```

		OFFC 00000		.ENTRY	ADDITIONAL OUTPUT, Save R2,R3,R4,R5,R6,R7,-	0518
		5B 00000000V	EF 9E 00002	MOVAB	R8,R9,R10,R11	
		5A 00000000V	EF 9E 00009	PUT_DESC, R11		
		59 00000000G	00 9E 00010	MOVAB	PUT-BLANK, R10	
		58 00000000G	00 9E 00017	MOVAB	DIF\$GL_REVADB+36, R9	
		57 00000000G	00 9E 0001E	MOVAB	DIF\$GL_MASADB+36, R8	
		5E 00000000G	10 C2 00025	MOVAB	DIF\$GL_FLAGS, R7	
		72 01	A7 E8 00028	SUBL2	#16, SP	
		6E 00000000'	EF 9B 0002C	BLBS	DIF\$GL_FLAGS+1, 68	0551
	04	AE 00000000'	EF 9E 00033	MOVZBW	STARS, STARDESC	0557
12	01	A7	05 E1 0003B	MOVAB	STARS+1, STARDESC+4	0558
5A		67	06 E0 00040	BBC	#5, DIF\$GL_FLAGS+1, 18	0564
56		67	05 E0 00044	BBS	#6, DIF\$GL_FLAGS, 68	0566
52		68	02 E0 00048	BBS	#5, DIF\$GL_FLAGS, 68	0567
4E		69	02 E0 0004C	BBS	#2, DIF\$GL_MASADB+36, 68	0568
			44 11 00050	BBS	#2, DIF\$GL_REVADB+36, 68	0569
31		67	06 E1 00052	BRB	48	0571
			56 D4 00056	BBC	#6, DIF\$GL_FLAGS, 28	0580
		51 00000000G	00 D0 00058	CLRL	FIRST	0582
	08	AE	51 B0 0005F	MOVL	DIF\$GL_PARWIDTH, R1	0583
		50 00000000G	00 D0 00063	MOVW	R1, DASHDESC	
				MOVL	DIF\$GL_OUTBUF, R0	0584

51	2D	OC	AE	50	DO	0006A	MOVL	R0, DASHDESC+4	0585	
			6E	00	2C	0006E	MOVCS	#0, (SP), #45, R1, (R0)		
				60		00073				
				08	AE	9F	PUSHAB	DASHDESC	0586	
			6B	01	FB	00077	CALLS	#1, PUT_DESC		
			6A	00	FB	0007A	CALLS	#0, PUT_BLANK	0587	
	20		68	03	EO	0007D	BBS	#3, DIF\$GL_MASFDB+36, 78	0588	
	05		69	03	EO	00081	BBS	#3, DIF\$GL_REVFDB+36, 38		
				17	11	00085	BRB	68	0589	
			56	01	DO	00087	28:	MOVL	#1, FIRST	0591
	13		68	03	EO	0008A	38:	BBS	#3, DIF\$GL_MASFDB+36, 78	0598
	0F		69	03	EO	0008E	BBS	#3, DIF\$GL_REVFDB+36, 78		
	05		67	03	EO	00092	BBS	#5, DIF\$GL_FLAGS, 58	0600	
				5E	DD	00096	48:	PUSHL	SP	0601
			68	01	FB	00098	CALLS	#1, PUT_DESC		
			6A	00	FB	0009B	58:	CALLS	#0, PUT_BLANK	0602
				0105	31	0009E	68:	BRW	338	0603
			54	67	DO	000A1	78:	MOVL	DIF\$GL_FLAGS, SAVE_FLAGS	0609
			52	01	DO	000A4	MOVL	#1, I	0614	
				53	D4	000A7	88:	CLRL	R3	0617
			01	52	D1	000A9	CMPL	I, #1		
				03	12	000AC	BNEQ	98		
				53	D6	000AE	INCL	R3		
			15	67	E8	000B0	BLBS	DIF\$GL_FLAGS, 138		
			02	52	D1	000B3	98:	CMPL	I, #2	0618
				04	12	000B6	BNEQ	108		
	OC		67	01	EO	000B8	BBS	#1, DIF\$GL_FLAGS, 138		
			03	52	D1	000BC	108:	CMPL	I, #3	0619
				03	13	000BF	BEQL	128		
				00AE	31	000C1	118:	BRW	298	
	F9		67	02	E1	000C4	128:	BBC	#2, DIF\$GL_FLAGS, 118	
			01	52	D1	000C8	138:	CMPL	I, #1	0622
				0A	13	000CB	BEQL	148		
			07	56	E8	000CD	BLBS	FIRST, 148		
	15	00000000V	EF	00	FB	000D0	CALLS	#0, INIT_HEX_OCTAL	0623	
			67	05	E1	000D7	148:	BBC	#5, DIF\$GL_FLAGS, 178	0638
			04	56	E9	000DB	BLBC	FIRST, 158	0640	
				56	D4	000DE	CLRL	FIRST	0641	
				08	11	000E0	BRB	168		
				20	88	000E2	158:	BISB2	#32, DIF\$GL_FLAGS+1	0643
		01	A7	00	FB	000E6	CALLS	#0, OUTPUT_MERGED	0644	
		00000000V	EF	00	FB	000ED	168:	CALLS	#0, PUT_BLANK	0646
			6A	02	E1	000F0	178:	BBC	#2, DIF\$GL_MASFDB+36, 208	0649
	1D		68	56	E9	000F4	BLBC	FIRST, 188	0651	
			04	56	D4	000F7	CLRL	FIRST	0652	
				0E	11	000F9	BRB	198		
		01	A7	20	88	000FB	188:	BISB2	#32, DIF\$GL_FLAGS+1	0654
		00000000V	EF	AB	9F	000FF	PUSHAB	DIF\$GL_MASFDB	0655	
				01	FB	00102	CALLS	#1, OUTPUT_SEPARATED		
				5E	DD	00109	198:	PUSHL	SP	0657
			6B	01	FB	0010B	CALLS	#1, PUT_DESC		
			6A	00	FB	0010E	CALLS	#0, PUT_BLANK	0658	
	1D		69	02	E1	00111	208:	BBC	#2, DIF\$GL_REVFDB+36, 238	0661
			04	56	E9	00115	BLBC	FIRST, 218	0663	
				56	D4	00118	CLRL	FIRST	0664	
				0E	11	0011A	BRB	228		
		01	A7	20	88	0011C	218:	BISB2	#32, DIF\$GL_FLAGS+1	0666

00000000V	EF	DC	A9	9F	00120	PUSHAB	DIF\$GL_REVFD8	0667
			01	FB	00123	CALLS	#1, OUTPUT_SEPARATED	0669
	6B		5E	DD	0012A	228: PUSHL	SP	0670
	6A		01	FB	0012C	CALLS	#1, PUT_DESC	0673
	1D		00	FB	0012F	CALLS	#0, PUT_BLANK	0675
	04		68	E9	00132	238: BLBC	DIF\$GL_MASFDB+36, 268	0676
			56	E9	00135	BLBC	FIRST, -248	0678
			56	D4	00138	CLRL	FIRST	0679
			0E	11	0013A	BRB	258	0681
01	A7		20	88	0013C	248: BISB2	#32, DIF\$GL_FLAGS+1	0682
00000000V	EF	DC	A8	9F	00140	PUSHAB	DIF\$GL_MASFDB	0683
			01	FB	00143	CALLS	#1, OUTPUT_CHANGEAB	0685
	6B		5E	DD	0014A	258: PUSHL	SP	0687
	6A		01	FB	0014C	CALLS	#1, PUT_DESC	0688
	1D		00	FB	0014F	CALLS	#0, PUT_BLANK	0690
	04		69	E9	00152	268: BLBC	DIF\$GL_REVFD8+36, 298	0691
			56	E9	00155	BLBC	FIRST, -278	0693
			56	D4	00158	CLRL	FIRST	0694
			0E	11	0015A	BRB	288	0702
01	A7		20	88	0015C	278: BISB2	#32, DIF\$GL_FLAGS+1	0703
00000000V	EF	DC	A9	9F	00160	PUSHAB	DIF\$GL_REVFD8	0704
			01	FB	00163	CALLS	#1, OUTPUT_CHANGEAB	0705
	6B		5E	DD	0016A	288: PUSHL	SP	0706
	6A		01	FB	0016C	CALLS	#1, PUT_DESC	0713
	05		00	FB	0016F	CALLS	#0, PUT_BLANK	0714
	67		53	E9	00172	298: BLBC	R3, 308	0715
			01	8A	00175	BICB2	#1, DIF\$GL_FLAGS	0717
			0D	11	00178	BRB	328	0718
	02		52	D1	0017A	308: CMPL	1, #2	0719
			05	12	0017D	BNEQ	318	0720
	67		02	8A	0017F	BICB2	#2, DIF\$GL_FLAGS	0721
			03	11	00182	BRB	328	0722
	67		04	8A	00184	318: BICB2	#4, DIF\$GL_FLAGS	0723
FF1A	52		03	F1	00187	328: ACBL	#3, #1, 1, 88	0724
67	01		54	F0	0018D	INSV	SAVE_FLAGS, #0, #1, DIF\$GL_FLAGS	0725
50	54		01	EF	00192	EXTZV	#1, #1, SAVE_FLAGS, R0	0726
67	01		50	F0	00197	INSV	R0, #1, #1, DIF\$GL_FLAGS	0727
50	54		02	EF	0019C	EXTZV	#2, #1, SAVE_FLAGS, R0	0728
67	01		50	F0	001A1	INSV	R0, #2, #1, DIF\$GL_FLAGS	0729
			01	D0	001A6	338: MOVL	#1, R0	0730
			04	001A9	RET			0731

; Routine Size: 426 bytes. Routine Base: \$CODE\$ + 0076

```
0719 1 GLOBAL ROUTINE output_listing_trailer =
0720 BEGIN
0721
0722 **
0723 FUNCTIONAL DESCRIPTION:
0724     Output the trailer for the listing.
0725
0726 INPUTS:
0727     None.
0728
0729 OUTPUTS:
0730     The trailer is output.
0731
0732 ROUTINE VALUES:
0733     Always true
0734
0735 --
0736 LOCAL
0737     linedesc : BBLOCK [dsc$c_s_bln],          ! Local string desc
0738     flag,                                         !
0739     mask,                                         ! mask
0740     outdesc : BBLOCK [dsc$c_s_bln];
0741
0742 cmd_bufpos = 0;                                ! initialize output routine
0743
0744 IF .dif$gl_flags [dif$u_slp]                    ! If SLP
0745 THEN BEGIN                                     ! Then just output slash
0746     linedesc [dsc$w_length] = 1;
0747     linedesc [dsc$a_pointer] = UPLIT('/');
0748     put_desc (linedesc);
0749     RETURN true;
0750 END;
0751
0752 Output the number of difference sections.
0753 output_cmdfao (difsec, .dif$gl_difsec);
0754 output_cmdentity (0);
0755
0756 Output the number of difference records.
0757 output_cmdfao (difrec, .dif$gl_difrec);
0758 output_cmdentity (0);
0759
0760 Skip a line and then output the command line.
0761
0762 put_blank ();
0763
0764 output_cmdcounted (cstring ('DIFFERENCES '));
```

```
477 0776 2
478 0777
479 0778
480 0779
481 0780
482 0781
483 0782
484 0783
485 0784
486 0785
487 0786
488 0787
489 0788
490 0789
491 0790
492 0791
493 0792
494 0793
495 0794
496 0795
497 0796
498 0797
499 0798
500 0799
501 0800
502 0801
503 0802
504 0803
505 0804
506 0805
507 0806
508 0807
509 0808
510 0809
511 0810
512 0811
513 0812
514 0813
515 0814
516 0815
517 0816
518 0817
519 0818
520 0819
521 0820
522 0821
523 0822
524 0823
525 0824
526 0825
527 0826
528 0827
529 0828
530 0829
531 0830
532 0831
533 0832

IF .dif$gl_flags [dif$u_ignore] ! /IGNORE
THEN
  BEGIN
  BIND
    list = PLIT (
      UPLIT (ign$u_blnklin, %ASCII 'BLANK LINES'),
      UPLIT (ign$u_comments, %ASCII 'COMMENTS'),
      UPLIT (ign$u_exact, %ASCII 'EXACT'),
      UPLIT (ign$u_formfeed, %ASCII 'FORM FEEDS'),
      UPLIT (ign$u_header, %ASCII 'HEADER'),
      UPLIT (ign$u_pretty, %ASCII 'PRETTY'),
      UPLIT (ign$u_traiblnk, %ASCII 'TRAILING SPACES'),
      UPLIT (ign$u_spacing, %ASCII 'SPACING'); : VECTOR [, LONG];
    mask = 0;
    DECR i FROM .list [-1] - 1 TO 0
    DO
      mask = .mask OR (..list [.i] AND .dif$gl_ignore);
    output_cmdcounted (cstring ('/IGNORE='));
    DECR i FROM .list [-1] - 1 TO 0
    DO
      IF (..list [.i] AND .mask) NEQ 0
      THEN
        BEGIN
          mask = .mask AND NOT ..list [.i];
          output_cmdcounted (.list [.i]+4);
          IF (..list [.i] AND ign$u_header) NEQ 0 ! special case HEADER
          THEN
            output_cmdfao (cstring ('=UL'), .dif$gl_header);
          IF .mask NEQ 0
          THEN
            output_cmdcounted (cstring (','));
          END;
        output_cmdcounted (cstring (')));
      END;
  IF .dif$gl_flags [dif$u_comdel] ! /COMMENT_DELIMITERS
  THEN
    BEGIN
      output_cmdcounted (cstring ('/COMMENT DELIMITERS'));
      IF .dif$gl_commdesc [dsc$u_length] NEQ 0
      THEN
        BEGIN
          output_cmdcounted (cstring ('='));
          INCR i FROM 0 TO .dif$gl_commdesc [dsc$u_length]-1
          DO
            BEGIN
              SELECT ONE CH$RCHAR (.dif$gl_commdesc [dsc$a_pointer]+.i) OF
              SET
                [X'00'] : output_cmdcounted (cstring ('COLON'));
                [X'01'] : output_cmdcounted (cstring ('COMMA'));
                [X'02'] : output_cmdcounted (cstring ('EXCLAMATION'));
                [X'03'] : output_cmdcounted (cstring ('FORM FEED'));
                [X'04'] : output_cmdcounted (cstring ('LEFT'));
                [X'05'] : output_cmdcounted (cstring ('RIGHT'));
                [X'06'] : output_cmdcounted (cstring ('SEMI COLON'));
                [X'07'] : output_cmdcounted (cstring ('SLASH'));
```



```
0833      [XC: '] : output_cmdcounted (cstring ('SPACE'));
0834      [XC: ] : output_cmdcounted (cstring ('TAB'));
0835      [OTHERWISE] :
0836      BEGIN
0837          outdesc [dsc$w_length] = 1;
0838          outdesc [dsc$a_pointer] = dif$gl_commdesc [dsc$a_pointer]+.1;
0839          output_cmdfao (cstring ('!AS'), outdesc);
0840      END;
0841      TES;
0842      IF .! NEQ .dif$gl_commdesc [dsc$w_length]-1
0843      THEN
0844          output_cmdcounted (cstring (','));
0845      END;
0846      output_cmdcounted (cstring ('')));
0847      END;
0848      END;
0849
0850      IF .dif$gl_flags [dif$v_width]          ! /WIDTH or /LINE_WIDTH
0851      THEN
0852          output_cmdfao (cstring ('/WIDTH=!UL'), .dif$gl_width);
0853
0854      IF .dif$gl_flags [dif$v_match]          ! /MATCH
0855      THEN
0856          output_cmdfao (cstring ('/MATCH=!UL'), .dif$gl_match);
0857
0858      IF .dif$gl_flags [dif$v_maxdif]         ! /MAXIMUM_DIFFERENCES
0859      THEN
0860          output_cmdfao (cstring ('/MAXIMUM_DIFFERENCES=!UL'), .dif$gl_maxdif);
0861
0862      IF .dif$gl_flags [dif$v_merged]         ! /MERGED
0863      THEN
0864          output_cmdfao (cstring ('/MERGED=!UL'), .dif$gl_merged);
0865
0866      IF NOT .dif$gl_flags [dif$v_ascii]      ! /MODE
0867      OR .dif$gl_flags [dif$v_hex]
0868      OR .dif$gl_flags [dif$v_octal]
0869      THEN
0870      BEGIN
0871      BIND
0872          list = PLIT (
0873              UPLIT (dif$m_ascii,          %ASCII 'ASCII'),
0874              UPLIT (dif$m_hex,           %ASCII 'HEXADECIMAL'),
0875              UPLIT (dif$m_octal,        %ASCII 'OCTAL')) : VECTOR [, LONG];
0876      mask = 0;
0877      DECR i FROM .list [-1] - 1 TO 0
0878      DO
0879          mask = .mask OR (..list [.i] AND .dif$gl_flags);
0880          output_cmdcounted (cstring ('/MODE=('));
0881          DECR i FROM .list [-1] - 1 TO 0
0882          DO
0883              IF (..list [.i] AND .mask) NEQ 0
0884              THEN
0885              BEGIN
0886                  mask = .mask AND NOT ..list [.i];
0887                  output_cmdcounted (..list [.i]+4);
0888                  IF .mask NEQ 0
0889                  THEN
```

```
591 0890      output_cmdcounted (cstring (''));
592 0891      END;
593 0892      output_cmdcounted (cstring (''));
594 0893      END;
595 0894
596 0895      IF .dif$gl_flags [dif$v_output]          ! /OUTPUT
597 0896      THEN
598 0897          BEGIN
599 0898              output_cmdcounted (cstring ('/OUTPUT='));
600 0899              output_cmdentity (dif$gl_outdesc);
601 0900              END;
602 0901
603 0902      IF .dif$gl_flags [dif$v_parallel]          ! /PARALLEL
604 0903      THEN
605 0904          output_cmdcounted (cstring ('/PARALLEL'));
606 0905
607 0906      IF .dif$gl_masfdb [fdb$v_separated] AND .dif$gl_revfdb [fdb$v_separated]
608 0907      THEN
609 0908          output_cmdcounted (cstring ('/SEPARATED'))
610 0909      ELSE IF .dif$gl_masfdb [fdb$v_separated]
611 0910      THEN
612 0911          output_cmdcounted (cstring ('/SEPARATED=MASTER'))
613 0912      ELSE IF .dif$gl_revfdb [fdb$v_separated]
614 0913      THEN
615 0914          output_cmdcounted (cstring ('/SEPARATED=REVISION'));
616 0915
617 0916      IF .dif$gl_flags [dif$v_slp]          ! /SLP
618 0917      THEN
619 0918          output_cmdcounted (cstring ('/SLP'));
620 0919
621 0920      IF .dif$gl_flags [dif$v_window]          ! /WINDOW
622 0921      THEN
623 0922          output_cmdfao (cstring ('/WINDOW=!UL'), .dif$gl_wndwsiz);
624 0923
625 0924      IF NOT .dif$gl_flags [dif$v_linenum]          ! /NUMBER
626 0925      THEN
627 0926          output_cmdcounted (cstring ('/NONUMBER'));
628 0927
629 0928      output_cmdcounted (cstring (''));          ! terminate line
630 0929
631 0930      output_cmdfile (dif$gl_masfdb);          ! output master file line
632 0931      output_cmdcounted (cstring (''));          ! terminate line
633 0932
634 0933      output_cmdfile (dif$gl_revfdb);          ! output revision file line
635 0934      output_cmdentity (0);          ! finish the output
636 0935
637 0936      ! put_desc (dif$gl_cmdesc);          ! original command line
638 0937
639 0938      RETURN true;
640 0939      END;
```

.PSECT SPLITS,NOWRT,NOEXE,2

```
00 00 00 2F 00000 P.AAA: .ASCII  \/\<0><0><0>
0C 00004 P.AAB: .BYTE  12
```

20	53	45	43	4E	45	52	45	46	46	49	44	000005	.ASCII	\DIFFERENCES \	:				
												00011	.BLKB	3	:				
												00014	P.AAD:	.LONG	1	:			
53	45	4E	49	4C	5F	4B	4E	41	4C	42	0B	00018	.ASCII	<11>\BLANK_LINES\	:				
												00024	P.AAE:	.LONG	2	:			
00	00	00	53	54	4E	45	4D	4D	4F	43	0B	00028	.ASCII	<8>\COMMENTS\<0><0><0>	:				
												00034	P.AAF:	.LONG	64	:			
				00	00	54	43	41	5B	45	05	00038	.ASCII	<5>\EXACT\<0><0>	:				
												00040	P.AAG:	.LONG	4	:			
00	53	44	45	45	46	5F	4D	52	4F	46	0A	00044	.ASCII	<10>\FORM_FEEDS\<0>	:				
												00050	P.AAH:	.LONG	32	:			
				00	52	45	44	41	45	48	06	00054	.ASCII	<6>\HEADER\<0>	:				
												0005C	P.AAI:	.LONG	128	:			
				00	59	54	54	45	52	50	06	00060	.ASCII	<6>\PRETTY\<0>	:				
												00068	P.AAJ:	.LONG	8	:			
45	43	41	50	53	5F	47	4E	49	4C	49	41	52	54	0F	0006C	.ASCII	<15>\TRAILING_SPACES\	:	
															0007B	:			
												0007C	P.AAK:	.LONG	16	:			
				47	4E	49	43	41	50	53	07	00080	.ASCII	<7>\SPACING\	:				
												00088	.LONG	8	:				
00000000'	00000000'	00000000'	00000000'	00000000'	00000000'	00000000'	00000000'	00000000'	00000000'	00000000'	00000000'	0008C	P.AAC:	.ADDRESS	P.AAD, P.AAE, P.AAF, P.AAG, P.AAH, - P.AAI, P.AAJ, P.AAK	:			
												000A4	P.AAL:	.BYTE	9	:			
				28	3D	45	52	4F	4E	47	49	2F	000AD	.ASCII	\IGNORE=(\	:			
												04	000B6	P.AAM:	.BYTE	4	:		
								4C	55	21	3D	000B7	.ASCII	\=!UL\	:				
												01	000BB	P.AAN:	.BYTE	1	:		
												2C	000BC	.ASCII	\,\	:			
												01	000BD	P.AAO:	.BYTE	1	:		
												29	000BE	.ASCII	\)\	:			
												13	000BF	P.AAP:	.BYTE	19	:		
49	4D	49	4C	45	44	5F	54	4E	45	4D	4D	4F	43	2F	000C0	.ASCII	\COMMENT_DELIMITERS\	:	
											53	52	45	54	000C1	:			
														02	000D3	P.AAQ:	.BYTE	2	:
												28	3D	000D4	.ASCII	\=(\	:		
														05	000D6	P.AAR:	.BYTE	5	:
								4E	4F	4C	4F	43	000D7	.ASCII	\COLON\	:			
												05	000DC	P.AAS:	.BYTE	5	:		
								41	4D	4D	4F	43	000DD	.ASCII	\COMMA\	:			
												0B	000E2	P.AAT:	.BYTE	11	:		
				4E	4F	49	54	41	4D	41	4C	43	5B	45	000E3	.ASCII	\EXCLAMATION\	:	
														09	000EE	P.AAU:	.BYTE	9	:
				44	45	45	46	5F	4D	52	4F	46	000EF	.ASCII	\FORM_FEED\	:			
												04	000F8	P.AAV:	.BYTE	4	:		
								54	46	45	4C	000F9	.ASCII	\LEFT\	:				
												05	000FD	P.AAW:	.BYTE	5	:		
								54	48	47	49	52	000FE	.ASCII	\RIGHT\	:			
												0A	00103	P.AAX:	.BYTE	10	:		
				4E	4F	4C	4F	43	5F	49	4D	45	53	00104	.ASCII	\SEMI_COLON\	:		
													05	0010E	P.AAY:	.BYTE	5	:	
								48	53	41	4C	53	0010F	.ASCII	\SLASH\	:			
												05	00114	P.AAZ:	.BYTE	5	:		
								45	43	41	50	53	00115	.ASCII	\SPACE\	:			
													03	0011A	P.ABA:	.BYTE	3	:	
										42	41	54	0011B	.ASCII	\TAB\	:			
												05	0011E	P.ABB:	.BYTE	5	:		
								22	53	41	21	22	0011F	.ASCII	\"!AS"! \	:			


```
01 00124 P.ABC: .BYTE 1
2C 00125 .ASCII \.
01 00126 P.ABD: .BYTE 1
29 00127 .ASCII \)
0A 00128 P.ABE: .BYTE 10
2F 00129 .ASCII \WIDTH=!UL\
0A 00133 P.ABF: .BYTE 10
2F 00134 .ASCII \MATCH=!UL\
18 0013E P.ABG: .BYTE 24
2F 0013F .ASCII \MAXIMUM_DIFFERENCES=!UL\
45 0014E
0B 00157 P.ABH: .BYTE 11
2F 00158 .ASCII \MERGED=!UL\
00163 .BLKB 1
00164 P.ABJ: .LONG 1
00168 .ASCII <5>\ASCII\<0><0>
00000002 00170 P.ABK: .LONG 2
4C 41 4D 49 43 45 44 41 58 45 48 0B 00174 .ASCII <11>\HEXADECIMAL\
00000004 00180 P.ABL: .LONG 4
00 00 4C 41 54 43 4F 05 00184 .ASCII <5>\OCTAL\<0><0>
00000003 0018C .LONG 3
00000000' 00000000' 00000000' 00190 P.ABI: .ADDRESS P.ABJ, P.ABK, P.ABL
07 0019C P.ABM: .BYTE 7
28 3D 45 44 4F 4D 2F 0019D .ASCII \MODE=(\
01 001A4 P.ABN: .BYTE 1
2C 001A5 .ASCII \.
01 001A6 P.ABO: .BYTE 1
29 001A7 .ASCII \)
0B 001A8 P.ABP: .BYTE 8
2F 001A9 .ASCII \OUTPUT=\
09 001B1 P.ABQ: .BYTE 9
2F 001B2 .ASCII \PARALLEL\
0A 001B8 P.ABR: .BYTE 10
2F 001BC .ASCII \SEPARATED\
11 001C6 P.ABS: .BYTE 17
2F 001C7 .ASCII \SEPARATED=MASTER\
45 001D6
13 001D8 P.ABI: .BYTE 19
2F 001D9 .ASCII \SEPARATED=REVISION\
53 001E8
04 001EC P.ABU: .BYTE 4
2F 001ED .ASCII \SLP\
0B 001F1 P.ABV: .BYTE 11
2F 001F2 .ASCII \WINDOW=!UL\
09 001FD P.ABW: .BYTE 9
2F 001FE .ASCII \NONUMBER\
00 00207 P.ABX: .BYTE 0
00208 .BLKB 0
00 00208 P.ABY: .BYTE 0
00209 .BLKB 0
```

```
LIST= P.AAC
LIST= P.ABI
```

```
.PSECT $CODE$,NOWRT,2
```

		OFFC 00000	.ENTRY	OUTPUT LISTING TRAILER, Save R2,R3,R4,R5,-		
	5B	00000000V	EF 9E 00002	MOVAB	R6,R7,R8,R9,R10,R11	0719
	5A	00000000V	EF 9E 00009	MOVAB	OUTPUT_CMDENTITY, R11	
	59	00000000G	00 9E 00010	MOVAB	OUTPUT_CMDFAO, R10	
	58	00000000V	EF 9E 00017	MOVAB	DIF\$GL_FLAGS, R9	
	57	00000000V	EF 9E 0001E	MOVAB	OUTPUT_CMDCOUNTED, R8	
	5E	00000000V	10 C2 00025	MOVAB	LIST, R7	
		00000000V	EF D4 00028	SUBL2	#16, SP	
	17	01	A9 E9 0002E	CLRL	CMD_BUFPOS	0747
OB	AE		01 B0 00032	BLBC	DIF\$GL_FLAGS+1, 1\$	0749
OC	AE	FF74	C7 9E 00036	MOVW	#1, LINEDESC	0751
		08	AE 9F 0003C	MOVAB	P.AAA, LINEDESC+4	0752
00000000CV	EF		01 FB 0003F	PUSHAB	LINEDESC	0753
		02BE	31 00046	CALLS	#1, PUT_DESC	
		00000000G	00 DD 00049	BRW	46\$	0754
		00000000V	EF 9F 0004F	PUSHL	DIF\$GL_DIFSEC	0760
	6A		02 FB 00055	PUSHAB	DIFSEC	
			7E D4 00058	CALLS	#2, OUTPUT_CMDFAO	
	6B		01 FB 0005A	CLRL	-(SP)	0761
		00000000G	00 DD 0005D	CALLS	#1, OUTPUT_CMDENTITY	
		00000000V	EF 9F 00063	PUSHL	DIF\$GL_DIFREC	0766
	6A		02 FB 00069	PUSHAB	DIFREC	
			7E D4 0006C	CALLS	#2, OUTPUT_CMDFAO	
	6B		01 FB 0006E	CLRL	-(SP)	0767
00000000V	EF		00 FB 00071	CALLS	#1, OUTPUT_CMDENTITY	
		FF7B	C7 9F 00078	CALLS	#0, PUT_BLANK	0773
	6B		01 FB 0007C	PUSHAB	P.AAB	0775
62	69		04 E1 0007F	CALLS	#1, OUTPUT_CMDCOUNTED	
			55 D4 00083	BBC	#4, DIF\$GL_FLAGS, 7\$	0777
	50	FC	A7 D0 00085	CLRL	MASK	0790
			12 11 00089	MOVL	LIST-4, 1	0791
	51		6740 D0 0008B	BRB	3\$	
		00000000G	00 D2 0008F	MOVL	LIST[1], R1	0793
51	61		52 CB 00096	MCOML	DIF\$GL_IGNORE, R2	
	55		51 CB 0009A	BICL3	R2, (RT), R1	
	EB		50 F4 0009D	BISL2	R1, MASK	
		20	A7 9F 000A0	SOBGEQ	1, 2\$	
	6B		01 FB 000A3	PUSHAB	P.AAL	0794
	52	FC	A7 D0 000A6	CALLS	#1, OUTPUT_CMDCOUNTED	
			30 11 000AA	MOVL	LIST-4, 1	0795
	50		6742 D0 000AC	BRB	6\$	
	55		60 D3 000B0	MOVL	LIST[1], R0	0797
			27 13 000B3	BITL	(R0), MASK	
	55		60 CA 000B5	BEQL	6\$	
		04	A0 9F 000B8	BICL2	(R0), MASK	0800
	6B		01 FB 000BB	PUSHAB	4(R0)	0801
	50		6742 D0 000BE	CALLS	#1, OUTPUT_CMDCOUNTED	
OC	60		05 E1 000C2	MOVL	LIST[1], R0	0802
		00000000G	00 DD 000C6	BBC	#5, (R0), 5\$	
		2A	A7 9F 000CC	PUSHL	DIF\$GL_HEADER	0804
	6A		02 FB 000CF	PUSHAB	P.AAM	
			55 D3 000D2	CALLS	#2, OUTPUT_CMDFAO	
			06 13 000D4	TSTL	MASK	0805
		2F	A7 9F 000D6	BEQL	6\$	
	6B		01 FB 000D9	PUSHAB	P.AAN	0807
CD			52 F4 000DC	CALLS	#1, OUTPUT_CMDCOUNTED	
				SOBGEQ	1, 4\$	0797

03

68	31	A7	9F	000DF	PUSHAB	P.AAO	0809
69		01	FB	000E2	CALLS	#1, OUTPUT_CMDCOUNTED	
		03	EO	000E5	78:	BBS	#3, DIF\$GL_FLAGS, 98
		00CE	31	000E9	88:	BRW	268
	33	A7	9F	000EC	98:	PUSHAB	P.AAP
68		01	FB	000EF	CALLS	#1, OUTPUT_CMDCOUNTED	0815
	00000000G	00	B5	000F2	TSTW	DIF\$GL_COMDESC	0816
		EF	13	000F8	BEQL	88	
	47	A7	9F	000FA	PUSHAB	P.AAO	0819
68		01	FB	000FD	CALLS	#1, OUTPUT_CMDCOUNTED	
56	00000000G	00	3C	00100	MOVZWL	DIF\$GL_COMDESC, R6	0820
54		01	CE	00107	MNEGL	#1, I	
	009D	31	0010A	BRW	238		
53	00000000G	00	00	0010D	108:	MOVL	DIF\$GL_COMDESC+4, R3
52		6443	9A	00114	MOVZBL	(1)[R3], R2	0823
3A		52	91	00118	CPB	R2, #58	0825
		05	12	0011B	BNEQ	118	
	4A	A7	9F	0011D	PUSHAB	P.AAR	
		5D	11	00120	BRB	208	
2C		52	91	00122	118:	CPB	R2, #44
		05	12	00125	BNEQ	128	0826
	50	A7	9F	00127	PUSHAB	P.AAS	
		53	11	0012A	BRB	208	
21		52	91	0012C	128:	CPB	R2, #33
		05	12	0012F	BNEQ	138	0827
	56	A7	9F	00131	PUSHAB	P.AAT	
		49	11	00134	BRB	208	
0C		52	91	00136	138:	CPB	R2, #12
		05	12	00139	BNEQ	148	0828
	62	A7	9F	0013B	PUSHAB	P.AAU	
		3F	11	0013E	BRB	208	
5B	8F	52	91	00140	148:	CPB	R2, #91
		05	12	00144	BNEQ	158	0829
	6C	A7	9F	00146	PUSHAB	P.AAV	
		34	11	00149	BRB	208	
5D	8F	52	91	0014B	158:	CPB	R2, #93
		05	12	0014F	BNEQ	168	0830
	71	A7	9F	00151	PUSHAB	P.AAW	
		29	11	00154	BRB	208	
3A		52	91	00156	168:	CPB	R2, #58
		05	12	00159	BNEQ	178	0831
	77	A7	9F	0015B	PUSHAB	P.AAX	
		1F	11	0015E	BRB	208	
2F		52	91	00160	178:	CPB	R2, #47
		06	12	00163	BNEQ	188	0832
	0082	C7	9F	00165	PUSHAB	P.AAY	
		14	11	00169	BRB	208	
20		52	91	0016B	188:	CPB	R2, #32
		06	12	0016E	BNEQ	198	0833
	008C	C7	9F	00170	PUSHAB	P.AAZ	
		09	11	00174	BRB	208	
09		52	91	00176	198:	CPB	R2, #9
		09	12	00179	BNEQ	218	0834
	008E	C7	9F	0017B	PUSHAB	P.ABA	
68		01	FB	0017F	208:	CALLS	#1, OUTPUT_CMDCOUNTED
		11	11	00182	BRB	228	
6E		01	B0	00184	218:	MOVW	#1, OUTDESC

04	AE	53	54	C1	00187	ADDL3	I, R3, OUTDESC+4	0838
			5E	DD	0018C	PUSHL	SP	0839
		0092	C7	9F	0018E	PUSHAB	P, ABB	
6A			02	FB	00192	CALLS	#2, OUTPUT_CMDFAO	
50		00000000G	00	3C	00195	MOVZWL	DIF\$GL_COMMDESC, R0	0842
			50	D7	0019C	DECL	R0	
50			54	D1	0019E	CMPL	I, R0	
		0098	07	13	001A1	BEQL	258	
			C7	9F	001A3	PUSHAB	P, ABC	0844
68			01	FB	001A7	CALLS	#1, OUTPUT_CMDCOUNTED	
54	02		56	F2	001AA	AOBLSS	R6, I, 248	0820
			03	11	001AE	BRB	258	
		009A	FF5A	31	001B0	BRW	108	
			C7	9F	001B3	PUSHAB	P, ABD	0846
68			01	FB	001B7	CALLS	#1, OUTPUT_CMDCOUNTED	
0D	01		01	E1	001BA	BBC	#1, DIF\$GL_FLAGS+1, 278	0850
		00000000G	00	DD	001BF	PUSHL	DIF\$GL_WIDTH	0852
		009C	C7	9F	001C5	PUSHAB	P, ABE	
6A			02	FB	001C9	CALLS	#2, OUTPUT_CMDFAO	
0D	01		02	E1	001CC	BBC	#2, DIF\$GL_FLAGS+1, 288	0854
		00000000G	00	DD	001D1	PUSHL	DIF\$GL_MATCH	0856
		00A7	C7	9F	001D7	PUSHAB	P, ABF	
6A			02	FB	001DB	CALLS	#2, OUTPUT_CMDFAO	
0D	01		06	E1	001DE	BBC	#6, DIF\$GL_FLAGS+1, 298	0858
		00000000G	00	DD	001E3	PUSHL	DIF\$GL_MAXDIF	0860
		00B2	C7	9F	001E9	PUSHAB	P, ABG	
6A			02	FB	001ED	CALLS	#2, OUTPUT_CMDFAO	
0D			05	E1	001FO	BBC	#5, DIF\$GL_FLAGS, 308	0862
		00000000G	00	DD	001F4	PUSHL	DIF\$GL_MERGED	0864
		00CB	C7	9F	001FA	PUSHAB	P, ABH	
6A			02	FB	001FE	CALLS	#2, OUTPUT_CMDFAO	
04			69	E9	00201	BLBC	DIF\$GL_FLAGS, 318	0866
53			01	E0	00204	BBS	#1, DIF\$GL_FLAGS, 318	0867
			02	E1	00208	BBC	#2, DIF\$GL_FLAGS, 368	0868
			55	D4	0020C	CLRL	MASK	0876
50		0100	C7	DD	0020E	MOVL	LIST-4, I	0877
			10	11	00213	BRB	358	
51		0104	C740	DD	00215	MOVL	LIST(I), R1	0879
			69	D2	0021B	MCOML	DIF\$GL_FLAGS, R2	
52			52	CB	0021E	BICL3	R2, (RT), R1	
61			51	C8	00222	BISL2	R1, MASK	
55			50	F4	00225	SOBGEQ	I, 328	
ED		0110	C7	9F	00228	PUSHAB	P, ABM	0880
68			01	FB	0022C	CALLS	#1, OUTPUT_CMDCOUNTED	
53		0100	C7	DD	0022F	MOVL	LIST-4, I	0881
			1F	11	00234	BRB	358	
50		0104	C743	DD	00236	MOVL	LIST(I), R0	0883
55			60	D3	0023C	BITL	(R0), MASK	
			14	13	0023F	BEQL	358	
55			60	CA	00241	BICL2	(R0), MASK	0886
	04		A0	9F	00244	PUSHAB	4(R0)	0887
68			01	FB	00247	CALLS	#1, OUTPUT_CMDCOUNTED	
			55	D3	0024A	TSTL	MASK	0888
			07	13	0024C	BEQL	358	
		0118	C7	9F	0024E	PUSHAB	P, ABN	0890
68			01	FB	00252	CALLS	#1, OUTPUT_CMDCOUNTED	
DE			53	F4	00255	SOBGEQ	I, 348	0883

		011A	C7	9F	00258	PUSHAB	P.ABO	0892
	68		01	FB	0025C	CALLS	#1, OUTPUT_CMDCOUNTED	0895
		01	A9	95	0025F	TSTB	DIF\$GL_FLAGS+1	0898
			10	18	00262	BGEQ	378	0899
		011C	C7	9F	00264	PUSHAB	P.ABP	0902
	68		01	FB	00268	CALLS	#1, OUTPUT_CMDCOUNTED	0904
		00000000G	00	9F	0026B	PUSHAB	DIF\$GL_OUTDESC	0906
	68		01	FB	00271	CALLS	#1, OUTPUT_CMDIDENTITY	0908
07	69		06	E1	00274	BBC	#6, DIF\$GL_FLAGS, 388	0909
		0125	C7	9F	00278	PUSHAB	P.ABQ	0911
	68		01	FB	0027C	CALLS	#1, OUTPUT_CMDCOUNTED	0912
50 00000000G	00		02	EF	0027F	EXT2V	#2, #1 DIF\$GL_MASFDB+36, R0	0914
	17		50	E9	00288	BLBC	R0, 408	0916
06 00000000G	00		02	E1	0028B	BBC	#2, DIF\$GL_REVFDDB+36, 398	0918
		012F	C7	9F	00293	PUSHAB	P.ABR	0920
			15	11	00297	BRB	418	0922
	06		50	E9	00299	BLBC	R0, 408	0924
		013A	C7	9F	0029C	PUSHAB	P.ABS	0926
			0C	11	002A0	BRB	418	0928
07 00000000G	00		02	E1	002A2	BBC	#2, DIF\$GL_REVFDDB+36, 428	0930
	68		C7	9F	002AA	PUSHAB	P.ABT	0931
	07	01	A9	E9	002B1	CALLS	#1, OUTPUT_CMDCOUNTED	0933
		0160	C7	9F	002B5	BLBC	DIF\$GL_FLAGS+1, 438	0934
	68		01	FB	002B9	PUSHAB	P.ABU	0936
0D 01	A9		03	E1	002BC	CALLS	#1, OUTPUT_CMDCOUNTED	0938
		00000000G	00	DD	002C1	BBC	#3, DIF\$GL_FLAGS+1, 448	0940
		0165	C7	9F	002C7	PUSHL	DIF\$GL_WNDQSI2	0942
	6A		02	FB	002CB	PUSHAB	P.ABV	0944
07 01	A9		04	E0	002CE	CALLS	#2, OUTPUT_CMDFAO	0946
		0171	C7	9F	002D3	BBS	#4, DIF\$GL_FLAGS+1, 458	0948
	68		01	FB	002D7	PUSHAB	P.ABW	0950
		017B	C7	9F	002DA	CALLS	#1, OUTPUT_CMDCOUNTED	0952
	68		01	FB	002DE	PUSHAB	P.ABX	0954
		00000000G	00	9F	002E1	CALLS	#1, OUTPUT_CMDCOUNTED	0956
00000000V	EF		01	FB	002E7	PUSHAB	DIF\$GL_MASFDB	0958
		017C	C7	9F	002EE	CALLS	#1, OUTPUT_CMDFILE	0960
	68		01	FB	002F2	PUSHAB	P.ABY	0962
		00000000G	00	9F	002F5	CALLS	#1, OUTPUT_CMDCOUNTED	0964
00000000V	EF		01	FB	002FB	PUSHAB	DIF\$GL_REVFDDB	0966
			7E	D4	00302	CALLS	#1, OUTPUT_CMDFILE	0968
	68		01	FB	00304	CLRL	-(SP)	0970
	50		01	DD	00307	CALLS	#1, OUTPUT_CMDIDENTITY	0972
			04	0030A	MOVL	#1, R0		0974
					RET			0976

; Routine Size: 779 bytes, Routine Base: \$CODE\$ + 0220

```
0940 ROUTINE output_cmdfile (fdb) =
0941 BEGIN
0942
0943 --
0944
0945 FUNCTIONAL DESCRIPTION:
0946
0947     Output a file line in the trailer of the listing.
0948
0949 INPUTS:
0950
0951     fdb = address of file descriptor block for file
0952
0953 OUTPUTS:
0954
0955     A file line of the trailer is output.
0956
0957 ROUTINE VALUES:
0958
0959     Always true
0960
0961 --
0962 MAP
0963     fdb : REF BBLOCK;                ! file descriptor block
0964
0965 LOCAL
0966     outdesc : BBLOCK [dsc$e_s_bln];
0967
0968 output_cmdentity (.fdb [fdb$b_l_fildesc]);
0969 IF .fdb [fdb$b_changebar]
0970 THEN
0971 BEGIN
0972     output_cmdcounted (cstring ('/CHANGE BAR'));
0973     IF (.fdb [fdb$b_linenum] XOR .dif$gl_flags [dif$b_linenum])
0974         OR .fdb [fdb$b_cbarchr] NEQ %C'!'
0975     THEN
0976         output_cmdcounted (cstring ('='));
0977     IF .fdb [fdb$b_cbarchr] NEQ %C'!'
0978     THEN
0979 BEGIN
0980     outdesc [dsc$w_length] = 1;
0981     outdesc [dsc$a_pointer] = fdb [fdb$b_cbarchr];
0982     output_cmdfao (cstring ('!!AS!!'), outdesc);
0983 END;
0984     IF (.fdb [fdb$b_linenum] XOR .dif$gl_flags [dif$b_linenum])
0985         AND .fdb [fdb$b_cbarchr] NEQ %C'!'
0986     THEN
0987         output_cmdcounted (cstring (''));
0988     IF .fdb [fdb$b_linenum] AND NOT .dif$gl_flags [dif$b_linenum]
0989     THEN
0990         output_cmdcounted (cstring ('NUMBER'));
0991     ELSE IF NOT .fdb [fdb$b_linenum] AND .dif$gl_flags [dif$b_linenum]
0992     THEN
0993         output_cmdcounted (cstring ('NONUMBER'));
0994     IF (.fdb [fdb$b_linenum] XOR .dif$gl_flags [dif$b_linenum])
0995         OR .fdb [fdb$b_cbarchr] NEQ %C'!'
0996     THEN
```



```
0997 3 output_cmdcounted (cstring ('')));
0998 3 END;
0999 3
1000 2 RETURN true;
1001 1 END;
```

```
52 41 42 5F 45 47 4E 41 4B 43 0B 00209 P.ABZ: .BYTE 11
2F 0020A .ASCII \CHANGE_BAR\
02 00215 P.ACA: .BYTE 2
3D 00216 .ASCII \=(\
05 00218 P.ACB: .BYTE 5
22 53 41 21 22 00219 .ASCII \'\AS'\
01 0021E P.ACC: .BYTE 1
2C 0021F .ASCII \.\
06 00220 P.ACD: .BYTE 6
52 45 42 4D 55 4E 00221 .ASCII \NUMBER\
08 00227 P.ACE: .BYTE 8
52 45 42 4D 55 4E 4F 4E 00228 .ASCII \NONUMBER\
01 00230 P.ACF: .BYTE 1
29 00231 .ASCII \\\
```

.PSECT \$CODE\$,NOWRT,2

```
00FC 00000 OUTPUT_CMDFILE:
57 00000000G 00 9E 00002 .WORD Save R2,R3,R4,R5,R6,R7
56 00000000V EF 9E 00009 MOVAB DIF$GL_FLAGS, R7
55 00000000' EF 9E 00010 MOVAB OUTPUT_CMDCOUNTED, R6
5E 0B C2 00017 MOVAB P.ABZ, R5
52 04 AC D0 0001A SUBL2 #8, SP
38 A2 DD 0001E MOVL FDB, R2
00000000V EF 01 FB 00021 PUSHL 56(R2)
03 24 A2 FB 00028 CALLS #1, OUTPUT_CMDIDENTITY
0086 31 0002C BLBS 36(R2), 18
55 DD 0002F BRW 118
66 01 FB 00031 18: PUSHL R5
01 01 EF 00034 CALLS #1, OUTPUT_CMDCOUNTED
01 04 EF 0003A EXTZV #1, #1, 36(R2), R3
50 53 C0 00040 EXTZV #4, #1, DIF$GL_FLAGS+1, R0
06 50 EB 00043 ADDL2 R3, R0
21 26 A2 91 00046 BLBS R0, 28
06 13 0004A CMPB 38(R2), #33
0C A5 9F 0004C BEQL 38
66 01 FB 0004F 28: PUSHAB P.ACA
21 26 A2 91 00052 38: CALLS #1, OUTPUT_CMDCOUNTED
16 13 00058 CLRL R4
54 D6 0005A CMPB 38(R2), #33
01 B0 0005C BEQL 48
04 AE 26 A2 9E 0005F INCL R4
5E DD 00064 MOVW #1, OUTDESC
MOVAB 38(R2), OUTDESC+4
PUSHL SP
```

50	01	A7	00000000V	EF	0F	A5	9F	00066	PUSHAB	P,ACB		
				01		02	FB	00069	CALLS	#2, OUTPUT_CMDFAO		
				50		04	EF	00070	EXTZV	#4, #1, DIF\$GL_FLAGS+1, R0	0984	
				09		53	CO	00076	ADDL2	R3, R0		
				06		50	E9	00079	BLBC	R0, 58		
						54	E9	0007C	BLBC	R4, 58	0985	
				66	15	A5	9F	0007F	PUSHAB	P,ACC	0987	
				0D		01	FB	00082	CALLS	#1, OUTPUT_CMDCOUNTED		
		05	01	A7		53	E9	00085	BLBC	R3, 78	0988	
						04	E0	00088	BBS	#4, DIF\$GL_FLAGS+1, 68		
					17	A5	9F	0008D	PUSHAB	P,ACD	0990	
				0B		0B	11	00090	BRB	88		
						53	E8	00092	BLBS	R3, 98	0991	
		06	01	A7		04	E1	00095	BBC	#4, DIF\$GL_FLAGS+1, 98		
					1E	A5	9F	0009A	PUSHAB	P,ACE	0993	
				66		01	FB	0009D	CALLS	#1, OUTPUT_CMDCOUNTED		
50	01	A7		01		04	EF	000A0	EXTZV	#4, #1, DIF\$GL_FLAGS+1, R0	0994	
				50		53	CO	000A6	ADDL2	R3, R0		
				03		50	E8	000A9	BLBS	R0, 108		
				06		54	E9	000AC	BLBC	R4, 118	0995	
					27	A5	9F	000AF	PUSHAB	P,ACF	0997	
				66		01	FB	000B2	CALLS	#1, OUTPUT_CMDCOUNTED		
				50		01	D0	000B5	MOVL	#1, R0	1000	
						04	000B8	RET			1001	

: Routine Size: 185 bytes, Routine Base: \$CODE\$ + 052B

```
705 1002 1 ROUTINE output_cmdfao (control, data1) =
706 1003 BEGIN
707 1004
708 1005 **
709 1006
710 1007 FUNCTIONAL DESCRIPTION:
711 1008
712 1009     Output part of trail using SFAO with one argument.
713 1010
714 1011 INPUTS:
715 1012
716 1013     control =      SFAO control string (counted string)
717 1014     data1  =      SFAO data item number 1
718 1015
719 1016 OUTPUTS:
720 1017
721 1018     Part of the trailer is output.
722 1019
723 1020 ROUTINE VALUES:
724 1021
725 1022     Always true
726 1023
727 1024 --
728 1025 MAP
729 1026     control : REF VECTOR [, BYTE];
730 1027
731 1028 LOCAL
732 1029     ctrdesc : BBLOCK [dsc$c_s_bln],          ! control string descriptor
733 1030     outdesc : BBLOCK [dsc$c_s_bln],
734 1031     tmpbuf  : VECTOR [nam$c_maxrss+dif$c_maxlisiz, BYTE];      ! formatting buffer
735 1032
736 1033     ctrdesc [dsc$w_length] = .control [0];
737 1034     ctrdesc [dsc$a_pointer] = control [1];
738 1035     outdesc [dsc$w_length] = %ALLOCATION (tmpbuf);
739 1036     outdesc [dsc$a_pointer] = tmpbuf;
740 1037     SYSSFAO (ctrdesc, outdesc [dsc$w_length], outdesc, .data1);
741 1038     output_cmdentity (outdesc);
742 1039
743 1040 RETURN true;
744 1041 END;
```

```
0000 00000 OUTPUT_CMDFAO:
FC AD          FB SE FE6C CE 9E 00002 .WORD Save nothing
          04 BC 9B 00007 MOVAB -404(SP), SP
          04 AC 01 C1 0000C MOVZBW @CONTROL, CTRDESC
          F0 AD 8F 80 00012 ADDL3 #1, CONTROL, CTRDESC+4
          F4 AD 6E 9E 00018 MOVW #387, OUTDESC
          08 AC DD 0001C MOVAB TMPBUF, OUTDESC+4
          F0 AD 9F 0001F PUSHL DATA1
          F0 AD 9F 00022 PUSHAB OUTDESC
          F8 AD 9F 00025 PUSHAB OUTDESC
          04 FB 00028 CALLS #4, SYSSFAO
00000000G 00
```

```
1002
1033
1034
1035
1036
1037
```


DIF OUTPUT
V04=000

⁶
15-Sep-1984 23:43:35
14-Sep-1984 12:19:23

VAX-11 B11ss-32 V4.0-742
DISK\$VMSMASTER:[DIF.SRC]OUTPUT.B32;1

Page 28
(7)

00000000V EF
50

FO	AD	9F	0002F
	01	FB	00032
	01	D0	00039
	04		0003C

PUSHAB	OUTDESC
CALLS	#1, OUTPUT_CMIDENTITY
MOVL	#1, R0
RET	

: 1038
: 1040
: 1041

; Routine Size: 61 bytes. Routine Base: \$CODE\$ + 05E4

D1
V0

```

746 1042 1 ROUTINE output_cmdcounted (str) =
747 1043 BEGIN
748 1044
749 1045 **
750 1046
751 1047 FUNCTIONAL DESCRIPTION:
752 1048
753 1049 Output a command line entity. Insert new
754 1050 lines as necessary.
755 1051
756 1052 INPUTS:
757 1053
758 1054 str = Counted string to output
759 1055
760 1056 OUTPUTS:
761 1057
762 1058 The entity is inserted in the output buffer. Output buffers
763 1059 are written as required.
764 1060
765 1061 IMPLICIT INPUTS/OUTPUTS:
766 1062
767 1063 cmd_bufpos = current position in output buffer
768 1064
769 1065 ROUTINE VALUES:
770 1066
771 1067 Always true.
772 1068
773 1069 --
774 1070 MAP
775 1071 str : REF VECTOR [, BYTE]; ! counted string
776 1072
777 1073 LOCAL
778 1074 outdesc : BBLOCK [dsc$a_s_bln]; ! string descriptor
779 1075
780 1076 outdesc [dsc$a_length] = .str [0];
781 1077 outdesc [dsc$a_pointer] = str [1];
782 1078 output_cmdentity (outdesc);
783 1079
784 1080 RETURN true;
785 1081 END;

```

				0000 00000 OUTPUT_CMDCOUNTED:				
			5E	08	C2 00002	WORD	Save nothing	1042
			6E	BC	9B 00005	SUBL2	#8, SP	1076
04	AE	04	AC	01	C1 00009	MOVZBW	%SI, OUTDESC	1077
			5E	DD	0000F	ADDL3	#1, %SI, OUTDESC+4	1078
	00000000v		EF	01	FB 00011	PUSHL	SP	1080
			50	01	D0 00018	CALLS	#1, OUTPUT_CMDENTITY	1081
					04 00018	MOVL	#1, R0	
						RET		

; Routine Size: 28 bytes, Routine Base: SCODES + 0621

DIF_OUTPUT
V04=000

N 6
15-Sep-1984 23:43:35
14-Sep-1984 12:19:23

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[DIF.SRC]OUTPUT.B32;1 Page 30
(8)

DI
VO


```
787 1082 1 ROUTINE output_cmdentity (descr) =
788 1083 BEGIN
789 1084
790 1085 ++
791 1086
792 1087 FUNCTIONAL DESCRIPTION:
793 1088
794 1089     Output a command line entity. Insert new
795 1090     lines as necessary.
796 1091
797 1092 INPUTS:
798 1093
799 1094     descr =      descriptor of entity to output
800 1095                If descr = 0, output buffer
801 1096                If length = 0, output buffer with continuation mark
802 1097                Otherwise, buffer and output data
803 1098
804 1099 OUTPUTS:
805 1100
806 1101     The entity is inserted in the output buffer. Output buffers
807 1102     are written as required.
808 1103
809 1104 IMPLICIT INPUTS/OUTPUTS:
810 1105
811 1106     cmd_bufpos =  current position in output buffer
812 1107
813 1108 ROUTINE VALUES:
814 1109
815 1110     Always true.
816 1111
817 1112 --
818 1113 MAP
819 1114     descr : REF BBLOCK;                ! string descriptor
820 1115
821 1116 LOCAL
822 1117     outdesc : BBLOCK [dsc$c_s_bln];    ! string descriptor
823 1118
824 1119 LITERAL
825 1120     indent = MINU (4, dif$c_minlisiz-1); ! indentation of continuation line
826 1121
827 1122 IF .descr EQL 0
828 1123 THEN
829 1124     BEGIN                                ! flush output buffer
830 1125         outdesc [dsc$w_length] = .cmd_bufpos;
831 1126         outdesc [dsc$a_pointer] = .dif$gl_outbuf;
832 1127         put_desc (outdesc);
833 1128         cmd_bufpos = 0;
834 1129     END
835 1130 ELSE IF .descr [dsc$w_length] EQL 0
836 1131 THEN
837 1132     BEGIN                                ! output line with continuation
838 1133         dif$gl_outbuf [.cmd_bufpos] = %C'-';
839 1134         cmd_bufpos = .cmd_bufpos+1;
840 1135         output_cmdentity T0);            ! output line
841 1136         CH$FILE (%C' ', indent, .dif$gl_outbuf);
842 1137         cmd_bufpos = indent;
843 1138     END
```

```
844 1139 2 ELSE IF .descr [dsc$w_length] LEQU .dif$gl_width-.cmd_bufpos-1
845 1140 THEN
846 1141 BEGIN ! fits on current line
847 1142 CHSMOVE (.descr [dsc$w_length], descr [dsc$a_pointer], dif$gl_outbuf [.cmd_bufpos]);
848 1143 cmd_bufpos = .cmd_bufpos+.descr [dsc$w_length];
849 1144 END
850 1145 ELSE IF .descr [dsc$w_length] LEQU .dif$gl_width-indent-1
851 1146 THEN
852 1147 BEGIN ! output line and put on new line
853 1148 outdesc [dsc$w_length] = 0;
854 1149 output_cmdentity (outdesc); ! output line with continuation
855 1150 output_cmdentity (.descr); ! put entity on new line
856 1151 END
857 1152 ELSE ! item too big for one line
858 1153 BEGIN
859 1154 IF .dif$gl_width-.cmd_bufpos-1 EQL 0
860 1155 THEN
861 1156 BEGIN
862 1157 outdesc [dsc$w_length] = 0;
863 1158 output_cmdentity (outdesc);
864 1159 END;
865 1160 outdesc [dsc$a_pointer] = .descr [dsc$a_pointer];
866 1161 WHILE true
867 1162 DO
868 1163 BEGIN
869 1164 outdesc [dsc$w_length] = MINU (
870 1165 .dif$gl_width-.cmd_bufpos-1,
871 1166 .descr [dsc$a_pointer]+.descr [dsc$w_length]-.outdesc [dsc$a_pointer]);
872 1167 output_cmdentity (outdesc);
873 1168 outdesc [dsc$a_pointer] = .outdesc [dsc$a_pointer]+.outdesc [dsc$w_length];
874 1169 IF .outdesc [dsc$a_pointer] EQLA .descr [dsc$a_pointer]+.descr [dsc$w_length]
875 1170 THEN
876 1171 EXITLOOP;
877 1172 outdesc [dsc$w_length] = 0;
878 1173 output_cmdentity (outdesc); ! output line with continuation
879 1174 cmd_bufpos = 0; ! no indentation
880 1175 END;
881 1176 END;
882 1177 RETURN true;
883 1178 END;
884 1179
```

OFFC 00000 OUTPUT_CMDENTITY:

5B	00000000G	00	9E	00002	WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	1082
5A	00000000G	00	9E	00009	MOVAB	DIF\$GL_WIDTH, R11	
59	ED	AF	9E	00010	MOVAB	DIF\$GL_OUTBUF, R10	
58	00000000'	EF	9E	00014	MOVAB	OUTPUT_CMDENTITY, R9	
5E		08	C2	0001B	MOVAB	CMD_BUFPOS, R8	
52		68	D0	0001E	SUBL2	#8, -SP	1125
56	04	AC	D0	00021	MOVL	CMD_BUFPOS, R2	1122
		14	12	00025	MOVL	DESCR, R6	
6E		52	B0	00027	BNEQ	1\$	
					MOVW	R2, OUTDESC	1125

04	AE	6A	D0	0002A	MOVL	DIF\$GL_OUTBUF, OUTDESC+4	1126
		5E	DD	0002E	PUSHL	SP	1127
00000000V	EF	01	FB	00030	CALLS	#1, PUT_DESC	
		68	D4	00037	CLRL	CMD_BUFPOS	1128
		58	11	00039	BRB	4\$	1122
		66	B5	0003B	TSTW	(R6)	1130
		1D	12	0003D	BNEQ	2\$	
50		6A	D0	0003F	MOVL	DIF\$GL_OUTBUF, R0	1133
6240		2D	90	00042	MOVB	#45, (R2)[R0]	
		68	D6	00046	INCL	CMD_BUFPOS	1134
		7E	D4	00048	CLRL	-(SP)	1135
		01	FB	0004A	CALLS	#1, OUTPUT_CMDENTITY	
		6A	D0	0004D	MOVL	DIF\$GL_OUTBUF, R0	1136
		8F	D0	00050	MOVL	#538978288, (R0)	
		04	D0	00057	MOVL	#4, CMD_BUFPOS	1137
		3A	11	0005A	BRB	4\$	1130
		68	D0	0005C	MOVL	DIF\$GL_WIDTH, R1	1139
57		51	C3	0005F	SUBL3	R2, R1, R7	
		50	A7	9E	MOVAB	-1(R7), R0	
50		66	00	ED	CMPZV	#0, #16, (R6), R0	
		10	11	1A	BGTRU	3\$	
		50	6A	D0	MOVL	DIF\$GL_OUTBUF, R0	1142
6240	04	86	66	28	MOVCL	(R6), #4(R6), (R2)[R0]	
		50	66	3C	MOVZWL	(R6), R0	1143
		68	50	C0	ADDL2	R0, CMD_BUFPOS	
			64	11	BRB	9\$	1139
50		50	A1	9E	MOVAB	-5(R1), R0	1145
		10	00	ED	CMPZV	#0, #16, (R6), R0	
			0E	1A	BGTRU	5\$	
			6E	B4	CLRW	OUTDESC	1148
			5E	DD	PUSHL	SP	1149
		69	01	FB	CALLS	#1, OUTPUT_CMDENTITY	
			56	DD	PUSHL	R6	1150
		69	01	FB	CALLS	#1, OUTPUT_CMDENTITY	
			4B	11	BRB	9\$	1145
		01	57	D1	CMPCL	R7, #1	1154
			07	12	BNEQ	6\$	
			6E	B4	CLRW	OUTDESC	1157
			5E	DD	PUSHL	SP	1158
		69	01	FB	CALLS	#1, OUTPUT_CMDENTITY	
51	04	AE	A6	D0	MOVL	4(R6), OUTDESC+4	1160
		68	68	C3	SUBL3	CMD_BUFPOS, DIF\$GL_WIDTH, R1	1165
			51	D7	DECL	R1	
		52	66	3C	MOVZWL	(R6), R2	1166
		52	A6	C0	ADDL2	4(R6), R2	
50		52	AE	C3	SUBL3	OUTDESC+4, R2, R0	
		50	51	D1	CMPCL	R1, R0	
			03	1B	BLEQU	8\$	
		51	50	D0	MOVL	R0, R1	
		6E	51	B0	MOVW	R1, OUTDESC	1164
			5E	DD	PUSHL	SP	1167
		69	01	FB	CALLS	#1, OUTPUT_CMDENTITY	
		50	6E	3C	MOVZWL	OUTDESC, R0	1168
	04	AE	50	C0	ADDL2	R0, OUTDESC+4	
		52	AE	D1	CMPCL	OUTDESC+4, R2	1169
			0B	13	BEQL	9\$	
			6E	B4	CLRW	OUTDESC	1172

69	5E	DD	000DA	PUSHL	SP	:	1173
	01	FB	000DC	CALLS	#1, OUTPUT_CMDENTITY	:	
	68	D4	000DF	CLRL	CMD_BUFPOS	:	1174
	C6	11	000E1	BRB	78	:	1161
50	01	D0	000E3	MOVL	#1, R0	:	1178
		04	000E6	RET		:	1179

: Routine Size: 231 Bytes, Routine Base: SCODES + 063D

.....

```
886 1180 1 ROUTINE output_changebar (fdb) =
887 1181 2 BEGIN
888 1182
889 1183 3 **
890 1184
891 1185 4 FUNCTIONAL DESCRIPTION:
892 1186
893 1187 5 This routine is called to output the most recent set of records,
894 1188 6 read from a particular input file, in CHANGEBAR format.
895 1189
896 1190 7 INPUTS:
897 1191
898 1192 8 fdb = The address of the FDB of the desired input file.
899 1193
900 1194 9 OUTPUTS:
901 1195
902 1196 10 The differences are written in changebar format to the output file.
903 1197
904 1198 11 ROUTINE VALUES:
905 1199
906 1200 12 Always true
907 1201
908 1202 13 --
909 1203
910 1204 14 MAP
911 1205 15 fdb : REF BBLOCK;
912 1206
913 1207 16 LOCAL
914 1208 17 cbarflag, ! Flag to output a change bar
915 1209 18 prevmatch, ! Flag is true if last record was a match
916 1210 19 rdb : REF BBLOCK; ! Address of the RDB of the current record
917 1211
918 1212 20 rdb = .fdb [fdb$l_firstdif]; ! Get first difference record
919 1213 21 prevmatch = true; ! Assume last record was a match
920 1214
921 1215 22 WHILE (.rdb NEQ .fdb [fdb$l_compnrec]) ! Output all unmatched records
922 1216 23 DO BEGIN
923 1217 24 IF .prevmatch AND .rdb [rdb$u_matchone] ! Output change bar if difference or if
924 1218 25 THEN cbarflag = 1 ! first match after a deletion
925 1219 26 ELSE cbarflag = (IF .rdb [rdb$u_match] THEN 2 ELSE 1);
926 1220 27 fdb [fdb$l_currec] = .rdb; ! Specify RDB of output record
927 1221 28 put_record (.fdb, .cbarflag); ! Output the record
928 1222 29 IF NOT .rdb [rdb$u_ignored] ! If record was not ignored
929 1223 30 THEN prevmatch = .rdb [rdb$u_match]; ! Update previous match flag
930 1224 31 rdb = .rdb [rdb$l_flink]; ! Get next record
931 1225 32 END;
932 1226
933 1227 33 IF .prevmatch AND .rdb [rdb$u_matchone] ! Output change bar if first match
934 1228 34 THEN cbarflag = 1 ! after a deletion
935 1229 35 ELSE cbarflag = (IF .rdb [rdb$u_match] THEN 2 ELSE 1);
936 1230 36 fdb [fdb$l_currec] = .rdb; ! Specify RDB of output record
937 1231 37 put_record (.fdb, .cbarflag); ! Output the record
938 1232
939 1233 38 RETURN true;
940 1234 39 END;
```

```
007C 00000 OUTPUT_CHANGE BAR:
      56 00000000V EF 9E 00002 .WORD Save R2,R3,R4,R5,R6      1180
      53 04 AC D0 00009 MOVAB PUT_RECORD, R6
      52 OC A3 D0 0000D MOVL FDB, R3      1212
      53 01 D0 00011 MOVL 12(R3), RDB
      14 A3 52 D1 00014 18: MOVL #1, PREVMATCH      1213
      53 2C 13 00018 CMPL RDB, 20(R3)      1215
      55 55 E9 0001A BEQL 68
      05 A2 05 E0 0001D BLBC PREVMATCH, 28      1217
      08 A2 04 E1 00022 BBS #5, 8(RDB), 38
      54 02 D0 00027 28: BBC #4, 8(RDB), 38      1219
      53 03 11 0002A MOVL #2, CBARFLAG
      54 01 D0 0002C BRB 48
      63 52 D0 0002F 38: MOVL #1, CBARFLAG
      18 BB 00032 48: MOVL RDB, (R3)      1220
      66 02 FB 00034 PUSHR #M<R3,R4>      1221
      06 A2 E8 00037 CALLS #2, PUT_RECORD
      55 01 04 EF 0003B BLBS 8(RDB), 58      1222
      52 62 D0 00041 58: EXTZV #4, #1, 8(RDB), PREVMATCH
      CE 11 00044 MOVL (RDB), RDB      1224
      55 55 E9 00046 18 BRB 18      1215
      05 A2 05 E0 00049 68: BLBC PREVMATCH, 78      1227
      08 A2 04 E1 0004E BBS #5, 8(RDB), 88
      54 02 D0 00053 78: BBC #4, 8(RDB), 88
      53 03 11 00056 MOVL #2, CBARFLAG
      54 01 D0 00058 BRB 98
      63 52 D0 0005B 88: MOVL #1, CBARFLAG
      18 BB 0005E 98: MOVL RDB, (R3)      1230
      66 02 FB 00060 PUSHR #M<R3,R4>      1231
      50 01 D0 00063 CALLS #2, PUT_RECORD
      04 00066 MOVL #1, R0      1233
      RET RET      1234
```

; Routine Size: 103 bytes. Routine Base: \$CODE\$ + 0724

```

942 1235 1 ROUTINE output_merged =
943 1236 1 BEGIN
944 1237 1
945 1238 1 **
946 1239 1
947 1240 1 FUNCTIONAL DESCRIPTION:
948 1241 1
949 1242 1     This routine is called to output the most recent set of detected
950 1243 1     differences in MERGED format.
951 1244 1
952 1245 1 IMPLICIT INPUTS:
953 1246 1
954 1247 1     The FDB's of the master and revision files.
955 1248 1
956 1249 1 OUTPUTS:
957 1250 1
958 1251 1     The differences are written to the output file in merged format.
959 1252 1
960 1253 1 ROUTINE VALUES:
961 1254 1
962 1255 1     Always true
963 1256 1
964 1257 1 --
965 1258 1
966 1259 1 LOCAL
967 1260 1     done,                                ! Flag is set when all differences have been output
968 1261 1     fdb : REF BBLOCK,                    ! Address of FDB of current output source
969 1262 1     rdb : REF BBLOCK,                    ! Address of RDB of current record
970 1263 1     stardesc : BBLOCK [dsc$c_s_bln];      ! Descriptor for line of stars
971 1264 1
972 1265 1 IF .dif$gl_flags [dif$u_init]              ! If init flag is set
973 1266 1 THEN dif$gl_flags [dif$u_init] = false; ! Then reset flag
974 1267 1
975 1268 1 done = false;                               ! Init until flag
976 1269 1 stardesc [dsc$a_pointer] = stars [1];       ! Init star desc address field
977 1270 1 stardesc [dsc$a_length] = .stars [0];      ! Init star desc length field
978 1271 1 dif$gl_masfdb [fdb$l_complrec] = .dif$gl_masfdb [fdb$l_firstdif]; ! Get first unmatched record
979 1272 1 dif$gl_revfdb [fdb$l_complrec] = .dif$gl_revfdb [fdb$l_firstdif]; ! from each input file
980 1273 1
981 1274 1 DO BEGIN                                     ! Loop until all differences output
982 1275 1
983 1276 1     fdb = dif$gl_masfdb;                       ! Output master file differences first
984 1277 1
985 1278 1     INCR 1 FROM 1 TO 2                         ! For both input files
986 1279 1     DO BEGIN                               ! Output differences
987 1280 1
988 1281 1         put_desc (stardesc);                   ! Output line of stars
989 1282 1         put_idline (.fdb);                     ! Output file id line
990 1283 1         rdb = .fdb [fdb$l_complrec];
991 1284 1
992 1285 1         WHILE ((NOT .rdb [rdb$u_matchone]) AND ! Output all unmatched records
993 1286 1             (NOT .rdb [rdb$u_eof]))
994 1287 1         DO BEGIN
995 1288 1             IF NOT .rdb [rdb$u_match]           ! If not a match record
996 1289 1             THEN BEGIN                       ! Then output it
997 1290 1                 fdb [fdb$l_currec] = .rdb;    ! Specify record to output
998 1291 1                 put_record (.fdb, 0);         ! Output record
```



```

999 1292 5
1000 1293
1001 1294
1002 1295
1003 1296
1004 1297
1005 1298
1006 1299
1007 1300
1008 1301
1009 1302
1010 1303
1011 1304
1012 1305
1013 1306
1014 1307
1015 1308
1016 1309
1017 1310
1018 1311
1019 1312
1020 1313
1021 1314
1022 1315
1023 1316
1024 1317
1025 1318
1026 1319
1027 1320
1028 1321
1029 1322
1030 1323
1031 1324
1032 1325
1033 1326
1034 1327
1035 1328
1036 1329
1037 1330
1038 1331

END;
rdb = .rdb [rdb$l_flink];
END;
INCR j FROM 1 TO .dif$gl_merged
DO BEGIN
    IF .rdb EQL .fdb [fdb$l_compnrec]
    THEN done = true;
    IF .rdb [rdb$u_ignored]
    THEN j = j-1
    ELSE BEGIN
        fdb [fdb$l_currec] = .rdb;
        put_record (.fdb, 0);
        END;
    rdb = .rdb [rdb$l_flink];
    END;
    IF .rdb EQL .fdb [fdb$l_compnrec]
    THEN done = true;
    IF .dif$gl_merged EQL 0
    THEN rdb = .rdb [rdb$l_flink];
    fdb [fdb$l_compnrec] = .rdb;
    fdb = dif$gl_revfdb;
    stardesc [dsc$u_length] = .stars [0] / 2;
    END;
    stardesc [dsc$u_length] = .stars [0];
    put_desc (stardesc);
END
UNTIL (.done);
RETURN true;
END;
```

! Get next record
! Output specified number of trailing matched records
! Check to see if no more differences
! No more, then set flag
! If ignore, then can't be a match
! Look again for a matched record
! Match, so output record
! Specify record to output
! Output record
! Get next record
! Check to see if no more differences
! No more, then set flag
! If no match records should be output
! Then skip one anyway
! Respecify first record of next potential difference section
! Get revision file fdb
! Use fewer stars
! Of increment
! Init star desc length field
! Output stars
! Of Until
! Of output_merged

```

OFFC 00000 OUTPUT_MERGED:
SB 00000000V EF 9E 00002 .WORD Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11 1235
SA 00000000V EF 9E 00009 MOVAB PUT_RECORD, R11
S9 00000000G 00 9E 00010 MOVAB PUT_DESC, R10
S8 00000000G 00 9E 00017 MOVAB DIF$GL_REV FDB+12, R9
S7 00000000' EF 9E 0001E MOVAB DIF$GL_MASFDB+12, R8
SE 00000000' 0C C2 00025 MOVAB STARS, R7
07 00000000G 00 05 E1 00028 SUBL2 #12, SP
00000000G 00 20 8A 00030 BBC #5, DIF$GL_FLAGS+1 18 1265
08 AE 01 A7 9E 00039 BICB2 #32, DIF$GL_FLAGS+1 1266
CLRL DONE 1268
MOVAB STARS+1, STARDESC+4 1269
```

	04	AE		67	9B	0003E	MOVZBW	STARS, STARDESC	1270
	04	A8		68	D0	00042	MOVL	DIF\$GL_MASFDB+12, DIF\$GL_MASFDB+16	1271
	04	A9		69	D0	00046	MOVL	DIF\$GL_REVFDDB+12, DIF\$GL_REVFDDB+16	1272
		53	F4	AB	9E	0004A	MOVAB	DIF\$GL_MASFDB, FDB	1276
		56		01	D0	0004E	MOVL	#1, I	1278
			04	AE	9F	00051	PUSHAB	STARDESC	1281
		6A		01	FB	00054	CALLS	#1, PUT_DESC	
				53	DD	00057	PUSHL	FDB	1282
00000000V		EF		01	FB	00059	CALLS	#1, PUT_IDLINE	
		52	10	A3	D0	00060	MOVL	16(FDB), RDB	1283
19	08	A2		03	E0	00064	BBS	#5, 8(RDB), 68	1285
14	08	A2		02	E0	00069	BBS	#2, 8(RDB), 68	1286
0A	08	A2		04	E0	0006E	BBS	#4, 8(RDB), 58	1288
		63		52	D0	00073	MOVL	RDB, (FDB)	1290
				7E	D4	00076	CLRL	-(SP)	1291
				53	DD	00078	PUSHL	FDB	
		68		02	FB	0007A	CALLS	#2, PUT_RECORD	
		52		62	D0	0007D	MOVL	(RDB), RDB	1293
				E2	11	00080	BRB	48	1285
		54	00000000G	00	D0	00082	MOVL	DIF\$GL_MERGED, R4	1296
				6E	D4	00089	CLRL	J	1299
				20	11	0008B	BRB	118	
		14	A3	52	D1	0008D	CMPL	RDB, 20(FDB)	
				03	12	00091	BNEQ	88	
		55		01	D0	00093	MOVL	#1, DONE	1300
		06	08	A2	E9	00096	BLBC	8(RDB), 98	1302
		6E	FF	AE	9E	0009A	MOVAB	J-1, J	1303
				0A	11	0009E	BRB	108	
		63		52	D0	000A0	MOVL	RDB, (FDB)	1305
				7E	D4	000A3	CLRL	-(SP)	1306
				53	DD	000A5	PUSHL	FDB	
		68		02	FB	000A7	CALLS	#2, PUT_RECORD	
		52		62	D0	000AA	MOVL	(RDB), RDB	1309
DC		6E		54	F3	000AD	AOBLEQ	R4, J, 78	1296
		14	A3	52	D1	000B1	CMPL	RDB, 20(FDB)	1312
				03	12	000B5	BNEQ	128	
		55		01	D0	000B7	MOVL	#1, DONE	1313
			00000000G	00	D5	000BA	TSTL	DIF\$GL_MERGED	1315
				03	12	000C0	BNEQ	138	
		52		62	D0	000C2	MOVL	(RDB), RDB	1316
		10	A3	52	D0	000C5	MOVL	RDB, 16(FDB)	1318
				53	A9	9E	MOVAB	DIF\$GL_REVFDDB, FDB	1320
		50	F4	67	9A	000CD	MOVZBL	STARS, R0	1321
		50		02	C6	000D0	DIVL2	#2, R0	
FF74		04	AE	50	B0	000D3	MOVW	R0, STARDESC	
		01		02	F1	000D7	ACBL	#2, #1, I, 38	1278
	56	04	AE	67	9B	000DD	MOVZBW	STARS, STARDESC	1324
				01	9F	000E1	PUSHAB	STARDESC	1325
		6A		01	FB	000E4	CALLS	#1, PUT_DESC	
		03		55	E8	000E7	BLBS	DONE, 128	1328
				FF5D	31	000EA	BRW	28	
		50		01	D0	000ED	MOVL	#1, R0	1330
				04	00	000F0	RET		1331

; Routine Size: 241 bytes, Routine Base: \$CODE\$ + 078B

```
1040 1332 1 ROUTINE output_parallel =
1041 1333 BEGIN
1042 1334
1043 1335 **
1044 1336
1045 1337 FUNCTIONAL DESCRIPTION:
1046 1338
1047 1339 This routine is called to output the most recent set of detected
1048 1340 differences in PARALLEL format. Note that unlike all the other output
1049 1341 routines, except SLP, this routine assumes it is being called on the
1050 1342 fly as difference sections are being discovered.
1051 1343
1052 1344 IMPLICIT INPUTS:
1053 1345
1054 1346 The FDB's of the master and revision files.
1055 1347
1056 1348 OUTPUTS:
1057 1349
1058 1350 The differences are written to the output file in parallel format.
1059 1351
1060 1352 ROUTINE VALUES:
1061 1353
1062 1354 Always true
1063 1355
1064 1356 --
1065 1357
1066 1358 LOCAL
1067 1359 linedesc : BBLOCK [dsc$_s_bln],
1068 1360 match,
1069 1361 masrdb : REF BBLOCK,
1070 1362 revrdb : REF BBLOCK;
1071 1363
1072 1364 IF .dif$gl_flags [dif$_init]
1073 1365 THEN BEGIN
1074 1366     dif$gl_flags [dif$_init] = false;
1075 1367     put_parallel_idline();
1076 1368 END;
1077 1369
1078 1370 masrdb = .dif$gl_masfdb [fdb$_firstdif];
1079 1371 revrdb = .dif$gl_revfdb [fdb$_firstdif];
1080 1372
1081 1373
1082 1374 Initialize the output descriptor and fill it with dashes.
1083 1375
1084 1376 linedesc [dsc$_length] = .dif$gl_parwidth;
1085 1377 linedesc [dsc$_pointer] = .dif$gl_outbuf;
1086 1378 CH$FILL (ASCII "-", linedesc [dsc$_length], linedesc [dsc$_pointer]);
1087 1379
1088 1380
1089 1381 If line numbers are requested, then insert them amidst the dashes.
1090 1382 Either way, output the line of dashes.
1091 1383
1092 1384 IF .dif$gl_flags [dif$_linenum]
1093 1385 THEN BEGIN
1094 1386     linedesc [dsc$_length] = .dif$gl_parwidth/ 4;
1095 1387     insert_linenum (.masrdb, linedesc, 1);
1096 1388     linedesc [dsc$_length] = 3 * .dif$gl_parwidth/ 4;
```



```
1097 1389      insert_linenum (.revrdb, linedesc, 1);
1098 1390      linedesc [dsc$u_length] = .dif$gl_parwidth;
1099 1391      END;
1100 1392      put_desc (linedesc);
1101 1393
1102 1394      ...
1103 1395      While the difference sections are of equal length, output the difference
1104 1396      records with text from both files.
1105 1397
1106 1398      WHILE (NOT .masrdb [rdb$u_matchone] AND NOT .revrdb [rdb$u_matchone]
1107 1399      AND NOT .masrdb [rdb$u_eof] AND NOT .revrdb [rdb$u_eof])
1108 1400      DO IF .masrdb [rdb$u_ignored]
1109 1401      THEN masrdb = .masrdb [rdb$l_flink]
1110 1402      ELSE IF .revrdb [rdb$u_ignored]
1111 1403      THEN revrdb = .revrdb [rdb$l_flink]
1112 1404      ELSE BEGIN
1113 1405          put_record_parallel (.masrdb, .revrdb);
1114 1406          masrdb = .masrdb [rdb$l_flink];
1115 1407          revrdb = .revrdb [rdb$l_flink];
1116 1408      END;
1117 1409
1118 1410      ...
1119 1411      While the master file difference section is longer than the revision
1120 1412      file difference section, output the difference records with text from
1121 1413      only the master file.
1122 1414
1123 1415      WHILE (NOT .masrdb [rdb$u_matchone] AND NOT .masrdb [rdb$u_eof])
1124 1416      DO BEGIN
1125 1417          IF NOT .masrdb [rdb$u_ignored]
1126 1418          THEN put_record_parallel (.masrdb, 0);
1127 1419          masrdb = .masrdb [rdb$l_flink];
1128 1420      END;
1129 1421
1130 1422      ...
1131 1423      While the revision file difference section is longer than the master
1132 1424      file difference section, output the difference records with text from
1133 1425      only the revision file.
1134 1426
1135 1427      WHILE (NOT .revrdb [rdb$u_matchone] AND NOT .revrdb [rdb$u_eof])
1136 1428      DO BEGIN
1137 1429          IF NOT .revrdb [rdb$u_ignored]
1138 1430          THEN put_record_parallel (0, .revrdb);
1139 1431          revrdb = .revrdb [rdb$l_flink];
1140 1432      END;
1141 1433
1142 1434      ...
1143 1435      Output matches from both files, until we are either done, or one of the
1144 1436      files runs out of records.
1145 1437
1146 1438      match = 0;
1147 1439      WHILE (.match NEQU .dif$gl_parallel) AND (NOT .masrdb [rdb$u_eof])
1148 1440      AND (NOT .revrdb [rdb$u_eof])
1149 1441      DO IF .masrdb [rdb$u_ignored]
1150 1442      THEN masrdb = .masrdb [rdb$l_flink]
1151 1443      ELSE IF .revrdb [rdb$u_ignored]
1152 1444      THEN revrdb = .revrdb [rdb$l_flink]
1153 1445      ELSE BEGIN
```



```
.. 1154
   1155
   1156
   1157
   1158
   1159
   1160
   1161

1446
1447
1448
1449
1450
1451
1452
1453
```

```
RETURN true;
END;
```

```
put_record_parallel (.masrdb, .revrdb);
masrdb = .masrdb [rdb$l_flink];
revrdb = .revrdb [rdb$l_flink];
match = .match + 1;
END;
```

! Of output_parallel

```
OFFC 00000 OUTPUT_PARALLEL:
      .WORD      Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11
      MOVAB      DIF$GL_FLAGS, R11
      MOVAB      DIF$GL_PARWIDTH, R10
      MOVAB      PUT_RECORD_PARALLEL, R9
      SUBL2      #8, -SP
      BBC        #5, DIF$GL_FLAGS+1, 1$
      BICB2      #32, DIF$GL_FLAGS+1
      CALLS      #0, PUT_PARALLEL_IDLINE
      MOVL       DIF$GL_MASFDB+12, MASRDB
      MOVL       DIF$GL_REVFDB+12, REVRDB
      MOVL       DIF$GL_PARWIDTH, R8
      MOVW       R8, LINEDESC
      MOVL       DIF$GL_OUTBUF, LINEDESC+4
      MOVCS      #0, (SP), #45, LINEDESC, @LINEDESC+4
      BBC        #4, DIF$GL_FLAGS+1, 2$
      DIVL3      #4, R8, R0
      MOVW       R0, LINEDESC
      PUSHL      #1
      PUSHAB     LINEDESC
      PUSHL      MASRDB
      CALLS      #3, INSERT_LINENUM
      MULL3      #3, DIF$GL_PARWIDTH, R0
      DIVL3      #4, R0, R1
      MOVW       R1, LINEDESC
      PUSHL      #1
      PUSHAB     LINEDESC
      PUSHL      REVRDB
      CALLS      #3, INSERT_LINENUM
      MOVW       DIF$GL_PARWIDTH, LINEDESC
      PUSHL      SP
      CALLS      #1, PUT_DESC
      BBS        #5, 8(MASRDB), 8$
      BBS        #5, 8(REVRDB), 6$
      BBS        #2, 8(MASRDB), 6$
      BBS        #2, 8(REVRDB), 6$
      BLBC       8(MASRDB), 4$
      MOVL       (MASRDB), MASRDB
      BRB        3$
      BLBS       8(REVRDB), 5$
      PUSHL      REVRDB
      PUSHL      MASRDB
      CALLS      #2, PUT_RECORD_PARALLEL

1332
1364
1366
1367
1370
1371
1376
1377
1378
1384
1386
1387
1388
1389
1390
1392
1398
1399
1400
1401
1402
1405
```

		57		67	D0	000B4		MOVL	(MASRDB), MASRDB	1406
		56		66	D0	000B7	5\$:	MOVL	(REVRDB), REVRDB	1407
				D0	11	000BA		BRB	3\$	1400
15	08	A7		05	E0	000BC	6\$:	BBS	#5, 8(MASRDB), 8\$	1415
10	08	A7		02	E0	000C1		BBS	#2, 8(MASRDB), 8\$	
		07	08	A7	E8	000C6		BLBS	8(MASRDB), 7\$	1417
				7E	D4	000CA		CLRL	-(SP)	1418
				57	DD	000CC		PUSHL	MASRDB	
		69		02	FB	000CE		CALLS	#2, PUT RECORD PARALLEL	
		57		67	D0	000D1	7\$:	MOVL	(MASRDBT), MASRDB	1419
				E6	11	000D4		BRB	6\$	1415
15	08	A6		05	E0	000D6	8\$:	BBS	#5, 8(REVRDB), 10\$	1427
10	08	A6		02	E0	000DB		BBS	#2, 8(REVRDB), 10\$	
		07	08	A6	E8	000E0		BLBS	8(REVRDB), 9\$	1429
				56	DD	000E4		PUSHL	REVRDB	1430
				7E	D4	000E6		CLRL	-(SP)	
		69		02	FB	000E8		CALLS	#2, PUT RECORD PARALLEL	
		56		66	D0	000EB	9\$:	MOVL	(REVRDBT), REVRDB	1431
				E6	11	000EE		BRB	8\$	1427
				52	D4	000F0	10\$:	CLRL	MATCH	1438
	00000000G	00		52	D1	000F2	11\$:	CMPL	MATCH, DIF\$GL_PARALLEL	1439
				2D	13	000F9		BEQL	14\$	
28	08	A7		02	E0	000FB		BBS	#2, 8(MASRDB), 14\$	
23	08	A6		02	E0	00100		BBS	#2, 8(REVRDB), 14\$	1440
		05	08	A7	E9	00105		BLBC	8(MASRDB), 12\$	1441
		57		67	D0	00109		MOVL	(MASRDB), MASRDB	1442
				E4	11	0010C		BRB	11\$	
		05	08	A6	E9	0010E	12\$:	BLBC	8(REVRDB), 13\$	1443
		56		66	D0	00112		MOVL	(REVRDB), REVRDB	1444
				DB	11	00115		BRB	11\$	
				56	DD	00117	13\$:	PUSHL	REVRDB	1446
				57	DD	00119		PUSHL	MASRDB	
		69		02	FB	0011B		CALLS	#2, PUT RECORD PARALLEL	
		57		67	D0	0011E		MOVL	(MASRDBT), MASRDB	1447
		56		66	D0	00121		MOVL	(REVRDB), REVRDB	1448
				52	D6	00124		INCL	MATCH	1449
				CA	11	00126		BRB	11\$	1441
		50		01	D0	00128	14\$:	MOVL	#1, R0	1452
				04	0012B			RET		1453

; Routine Size: 300 bytes, Routine Base: \$CODE\$ + 087C

```
1163 1454 1 ROUTINE output_separated (fdb) =
1164 1455 BEGIN
1165 1456
1166 1457
1167 1458
1168 1459 FUNCTIONAL DESCRIPTION:
1169 1460
1170 1461 This routine is called to output the most recent set of detected
1171 1462 differences from a particular input file in SEPARATED format.
1172 1463
1173 1464 INPUTS:
1174 1465
1175 1466 fdb = The address of the FDB of the desired input file.
1176 1467
1177 1468 OUTPUTS:
1178 1469
1179 1470 The differences are written to the separated output file.
1180 1471
1181 1472 ROUTINE VALUES:
1182 1473
1183 1474 Always true
1184 1475
1185 1476 --
1186 1477
1187 1478 MAP
1188 1479 fdb : REF BBLOCK;
1189 1480
1190 1481 LOCAL
1191 1482 rdb : REF BBLOCK, ! Address of the RDB of the current record
1192 1483 stardesc : BBLOCK [dsc$_s_bln]; ! Descriptor for stars
1193 1484
1194 1485 IF .dif$_flags [dif$_init] ! If init flag is set
1195 1486 THEN BEGIN ! Then output listing header
1196 1487 dif$_flags [dif$_init] = false; ! And reset flag
1197 1488 stardesc [dsc$_length] = .stars [0];
1198 1489 stardesc [dsc$_pointer] = stars [1];
1199 1490 put_desc (stardesc);
1200 1491 put_idline (.fdb);
1201 1492 END;
1202 1493
1203 1494 rdb = .fdb [fdb$_firstdif]; ! Get first difference record
1204 1495
1205 1496 WHILE (.rdb NEQ .fdb [fdb$_compnrec]) ! Output all unmatched records
1206 1497 DO BEGIN
1207 1498 IF NOT .rdb [rdb$_match] ! If difference
1208 1499 THEN BEGIN ! Then, output it
1209 1500 fdb [fdb$_currec] = .rdb; ! Specify output record
1210 1501 put_record (.fdb, 0); ! Output the record
1211 1502 END;
1212 1503 rdb = .rdb [rdb$_flink]; ! Get the next record
1213 1504
1214 1505 END;
1215 1506 RETURN true;
1216 1507 END;
```

```
000C 00000 OUTPUT_SEPARATED:
      5E      08 C2 00002      .WORD      Save R2,R3      : 1454
29 00000000G 00      05 E1 00005      SUBL2      #8, SP      : 1485
      00000000G 00      20 8A 0000D      BBC      #5, DIF$GL_FLAGS+1, 1$ : 1487
      04 6E 00000000' EF 9B 00014      BICB2      #32, DIF$GC_FLAGS+1 : 1488
      AE 00000000' EF 9E 0001B      MOVZBW      STARS, STARDESC : 1489
      00000000V EF      SE DD 00023      MOVAB      STARS+1, STARDESC+4 : 1490
      04      01 FB 00025      PUSHL      SP : 1491
      00000000V EF      AC DD 0002C      CALLS      #1, PUT_DESC : 1491
      52      04      01 FB 0002F      PUSHL      FDB : 1491
      53      04      AC D0 00036 1$:      CALLS      #1, PUT_IDLINE : 1494
      14 A2      0C      A2 D0 0003A      MOVL      FDB, R2 : 1494
      OE 08 A3      53 D1 0003E 2$:      MOVL      12(R2), RDB : 1496
      62      18 13 00042      BEQL      4$ : 1498
      00000000V EF      04 E0 00044      BBS      #4, 8(RDB), 3$ : 1498
      53      53 D0 00049      MOVL      RDB, (R2) : 1500
      7E D4 0004C      CLRL      -(SP) : 1501
      52 DD 0004E      PUSHL      R2 : 1501
      02 FB 00050      CALLS      #2, PUT_RECORD : 1503
      63 D0 00057 3$:      MOVL      (RDB), RDB : 1496
      E2 11 0005A      BRB      2$ : 1506
      01 D0 0005C 4$:      MOVL      #1, R0 : 1506
      04 0005F      RET : 1507
```

; Routine Size: 96 bytes. Routine Base: \$CODE\$ + 09A8


```
1218 1508 1 ROUTINE output_slp =
1219 1509 BEGIN
1220 1510
1221 1511 ++
1222 1512
1223 1513 FUNCTIONAL DESCRIPTION:
1224 1514
1225 1515 This routine is called to output the most recent edit to the
1226 1516 SLP output file. Unlike all the other output routine, except
1227 1517 PARALLEL, this routine assumes that it is being called on the
1228 1518 fly, as each new difference section is detected.
1229 1519
1230 1520 IMPLICIT INPUTS:
1231 1521
1232 1522 The FDB's of the two input files.
1233 1523
1234 1524 OUTPUTS:
1235 1525
1236 1526 The edits are written to the output file in SLP format.
1237 1527
1238 1528 ROUTINE VALUES:
1239 1529
1240 1530 Always true
1241 1531
1242 1532 --
1243 1533
1244 1534 LOCAL
1245 1535 charptr, ! Pointer into output buffer
1246 1536 linedesc : BBLOCK [dsc$c_s_bln], ! Descriptor of output line
1247 1537 number, ! Record number
1248 1538 rdb : REF BBLOCK; ! Address of the RDB of the current record
1249 1539
1250 1540 BIND
1251 1541 firstrdb = .dif$gl_masfdb [fdb$l_firstdif] : BBLOCK, ! RDB of first record in difference section
1252 1542 compnrdb = .dif$gl_masfdb [fdb$l_compnrdb] : BBLOCK; ! RDB of first match after difference section
1253 1543
1254 1544 IF .dif$gl_flags [dif$v_init] ! If init flag is set
1255 1545 THEN dif$gl_flags [dif$v_init] = false; ! Then reset flag
1256 1546
1257 1547 charptr = .dif$gl_outbuf; ! Init ptr to output buffer
1258 1548 CH$WCHAR_A (XC'- ', charptr); ! Insert '-' in output buffer
1259 1549 linedesc [dsc$b_class] = dsc$b_class_s; ! Set desc class for RTL routine
1260 1550
1261 1551 IF firstrdb EQL compnrdb ! Are we deleting any lines?
1262 1552 THEN number = .firstrdb [rdb$l_number] ! No, then get starting point of insertion
1263 1553 ELSE BEGIN ! Yes, then we must be replacing lines
1264 1554 linedesc [dsc$b_length] = dif$c_linenum - 1; ! Put number of first line to replace into
1265 1555 linedesc [dsc$b_pointer] = charptr; ! the output buffer
1266 1556 OT$SCVT_L_I (firstrdb [rdb$l_number], linedesc);
1267 1557 charptr = charptr + dif$c_linenum - 1;
1268 1558 number = .compnrdb [rdb$l_number]; ! Get number of line after last line to replace
1269 1559 END;
1270 1560
1271 1561 IF (number = .number - 1) NEQ 0 ! Calculate number of last line to replace
1272 1562 ! or point of insertion
1273 1563 THEN BEGIN ! If not at beginning of file
1274 1564
```

```
1275 1565 3 IF firstbdb NEQ compnrb;
1276 1566 THEN CH$CHAR A (ZC, charptr);
1277 1567 linedesc [dsc$w_length] = dif$gl_linenum - 1;
1278 1568 linedesc [dsc$a_pointer] = charptr;
1279 1569 OT$SCVT_L_T1 (number, linedesc);
1280 1570 charptr = charptr + dif$gl_linenum - 1;
1281 1571 END;
1282 1572
1283 1573 linedesc [dsc$a_pointer] = dif$gl_outbuf;
1284 1574 linedesc [dsc$w_length] = charptr - dif$gl_outbuf;
1285 1575 put_desc (linedesc);
1286 1576
1287 1577 rdb = dif$gl_revfdb [fdb$l_firstdif];
1288 1578
1289 1579 WHILE (.rdb NEQ dif$gl_revfdb [fdb$l_compnrec])
1290 1580 DO BEGIN
1291 1581 dif$gl_revfdb [fdb$l_currec] = rdb;
1292 1582 put_record (dif$gl_revfdb, 0);
1293 1583 rdb = rdb [rdb$l_link];
1294 1584 END;
1295 1585
1296 1586 RETURN true;
1297 1587 1 END;
```

```
00FC 00000 OUTPUT_SLP:
      57 00000000G 00 9E 00002 .WORD Save R2,R3,R4,R5,R6,R7
      56 00000000G 00 9E 00009 MOVAB OT$SCVT_L_T1, R7
      55 00000000G 00 9E 00010 MOVAB DIF$GL_OUTBUF, R6
      5E 0C C2 00017 MOVAB DIF$GL_REVFDB, R5
      52 00000000G 00 D0 0001A SUBL2 #12, SP
      53 00000000G 00 D0 00021 MOVL DIF$GL_MASFDB+12, R2
      07 00000000G 00 05 E1 00028 MOVL DIF$GL_MASFDB+20, R3
      00000000G 00 20 8A 00030 BBC #5, DIF$GL_FLAGS+1, 1$
      54 66 D0 00037 BICB2 #32, DIF$GL_FLAGS+1
      84 2D 90 0003A MOVL DIF$GL_OUTBUF, CHARPTR
      07 AE 01 90 0003D MOVAB #45, (CHARPTR)+
      53 52 D1 00041 MOVAB #1, LINEDESC+3
      06 12 00044 CMPL R2, R3
      6E 04 A2 D0 00046 BNEQ 2$
      04 AE 05 B0 0004C BRB 3$
      08 AE 54 D0 00050 MOVW #5, LINEDESC
      04 AE 9F 00054 MOVL CHARPTR, LINEDESC+4
      04 A2 9F 00057 PUSHAB LINEDESC
      67 02 FB 0005A PUSHAB 4(R2)
      54 05 C0 0005D CALLS #2, OT$SCVT_L_T1
      6E 04 A3 D0 00060 ADDL2 #5, CHARPTR
      1C 13 00066 MOVL 4(R3), NUMBER
      53 52 D1 00068 DECL NUMBER
      84 2C 90 0006D BEQL 5$
      BEQL 4$
      BEQL 4$
      MOVAB #44, (CHARPTR)+
```

04	AE	05	B0	00070	48:	MOVW	#5, LINEDESC	1567
08	AE	54	D0	00074		MOVL	CHARPTR, LINEDESC+4	1568
		04	AE	9F	00078	PUSHAB	LINEDESC	1569
		04	AE	9F	00078	PUSHAB	NUMBER	
	67	05	FB	0007E		CALLS	#2, OTSSCVT_L_T1	
	54	05	CO	00081		ADDL2	#5, CHARPTR	1570
	50	66	D0	00084	58:	MOVL	DIF\$GL_OUTBUF, R0	1573
04	AE	50	D0	00087		MOVL	R0, LINEDESC+4	
	54	50	A3	00088		SUBW3	R0, CHARPTR, LINEDESC	1574
		04	AE	9F	00090	PUSHAB	LINEDESC	1575
	00000000V	01	FB	00093		CALLS	#1, PUT_DESC	
		0C	A5	D0	0009A	MOVL	DIF\$GL_REVFDDB+12, RDB	1577
	14	52	D1	0009E	68:	CMPL	RDB, DIF\$GL_REVFDDB+20	1579
		13	13	000A2		BEQL	78	
	65	52	D0	000A4		MOVL	RDB, DIF\$GL_REVFDDB	1581
		7E	D4	000A7		CLRL	-(SP)	1582
		55	DD	000A9		PUSHL	R5	
	00000000V	02	FB	000AB		CALLS	#2, PUT_RECORD	1583
		62	D0	000B2		MOVL	(RDB), RDB	1579
		E7	11	000B5		BRB	68	
	50	01	D0	000B7	78:	MOVL	#1, R0	1586
		04	000BA			RET		1587

; Routine Size: 187 bytes, Routine Base: \$CODE\$ + 0A08

```
1299 1588 1 GLOBAL ROUTINE put_record (fdb, cbarflag) =
1300 1589 BEGIN
1301 1590
1302 1591 ++
1303 1592
1304 1593 FUNCTIONAL DESCRIPTION:
1305 1594
1306 1595     Call the appropriate radix output routine to format and put a
1307 1596     record to the output file.
1308 1597
1309 1598 INPUTS:
1310 1599
1311 1600     fdb =       The address of the FDB pointing to the CURREC that is
1312 1601             to be output.
1313 1602
1314 1603     cbarflag =  A flag that is 0 if not changebar format
1315 1604             1 if changebar format and bar should be output
1316 1605             2 if changebar format and bar should not be output
1317 1606
1318 1607 OUTPUTS:
1319 1608
1320 1609     The CURREC is output.
1321 1610
1322 1611 ROUTINE VALUES:
1323 1612
1324 1613     Always true
1325 1614
1326 1615 --
1327 1616 MAP
1328 1617     fdb : REF BBLOCK;
1329 1618
1330 1619 LOCAL
1331 1620     rdb : REF BBLOCK,
1332 1621     stardesc : BBLOCK [dsc&c_s_bln];
1333 1622
1334 1623
1335 1624     If init flag is set, then this must be a change bar listing. Output header
1336 1625     here, instead of in OUTPUT CHANGEBAR, because we can never tell if the first
1337 1626     record output will be a match or a difference.
1338 1627
1339 1628     IF .dif&gl_flags [dif&v_init]
1340 1629     THEN BEGIN
1341 1630         dif&gl_flags [dif&v_init] = false;
1342 1631         stardesc [dsc&w_length] = .stars [0];
1343 1632         stardesc [dsc&a_pointer] = stars [1];
1344 1633         put_desc (stardesc);
1345 1634         put_idline (.fdb);
1346 1635     END;
1347 1636
1348 1637     rdb = .fdb [fdb&l_currec];
1349 1638
1350 1639     IF .rdb [rdb&v_eof] OR .rdb [rdb&v_ignored]
1351 1640     THEN RETURN true;
1352 1641
1353 1642     IF .dif&gl_flags [dif&v_ascii]
1354 1643     THEN put_record_ascii (.fdb, .cbarflag)
1355 1644     ELSE put_record_hex_octal (.fdb, .cbarflag);
```

! If init flag is set
! Then output listing header
! And reset flag

! Get address of RDB of record to output

! If EOF or ignore
! Then don't output, simply return

! Call appropriate mode record output routine


```

: 1356
: 1357
: 1358
1645 2
1646 2 RETURN true;
1647 1 END;

```

				0004 00000	.ENTRY	PUT RECORD, Save R2	1588
		52	00000000G	00 9E 00002	MOVAB	DIF\$GL_FLAGS, R2	
		5E		08 C2 00009	SUBL2	#8, SP	
26	01	A2		05 E1 0000C	BBC	#5, DIF\$GL_FLAGS+1, 18	1628
	01	A2		20 8A 00011	BICB2	#32, DIF\$GL_FLAGS+1	1630
		6E	00000000'	EF 9B 00015	MOVZBW	STAR\$, STARDESC	1631
	04	AE	00000000'	EF 9E 0001C	MOVAB	STAR\$+1, STARDESC+4	1632
				5E DD 00024	PUSHL	SP	1633
		EF		01 FB 00026	CALLS	#1, PUT_DESC	
			04	AC DC 0002D	PUSHL	FDB	1634
		EF		01 FB 00030	CALLS	#1, PUT_IDLINE	
		50		BC D0 00037	MOVL	@FDB, RDB	1637
1F	08	A0		02 E0 0003B	BBS	#2, 8(RDB), 38	1639
		1B		A0 E8 00040	BLBS	8(RDB), 38	
		0D		62 E9 00044	BLBC	DIF\$GL_FLAGS, 28	1642
		7E		AC 7D 00047	MOVQ	FDB, -(SP)	1643
		EF		02 FB 0004B	CALLS	#2, PUT_RECORD_ASCII	
			04	0B 11 00052	BRB	38	
		7E		AC 7D 00054	MOVQ	FDB, -(SP)	1644
		EF		02 FB 00058	CALLS	#2, PUT_RECORD_HEX_OCTAL	
		50		01 D0 0005F	MOVL	#1, R0	1646
				04 00062	RET		1647

; Routine Size: 99 bytes. Routine Base: \$CODE\$ + 0AC3

```
1360 1648 1 ROUTINE put_record_ascii (fdb, cbarflag) =
1361 1649 BEGIN
1362 1650
1363 1651 **
1364 1652
1365 1653 FUNCTIONAL DESCRIPTION:
1366 1654
1367 1655     format and put a record to the output file in ascii mode.
1368 1656
1369 1657 INPUTS:
1370 1658
1371 1659     fdb =       The address of the FDB pointing to the CURREC that is
1372 1660             to be output.
1373 1661
1374 1662     cbarflag =  A flag that is 0 if not changebar format
1375 1663             1 if changebar format and bar should be output
1376 1664             2 if changebar format and bar should not be output
1377 1665
1378 1666 OUTPUTS:
1379 1667
1380 1668     The CURREC is output.
1381 1669
1382 1670 ROUTINE VALUES:
1383 1671
1384 1672     Always true
1385 1673
1386 1674 --
1387 1675
1388 1676 MAP
1389 1677     fdb : REF BBLOCK;
1390 1678
1391 1679 LOCAL
1392 1680     charptr,                               ! Pointer into the output buffer
1393 1681     linedesc : BBLOCK [dsc$e_s_bln],       ! Descriptor of output line
1394 1682     rdb : REF BBLOCK,                     ! Address of the RDB of the record to be output
1395 1683     text_length,                          ! Space in record for text
1396 1684     status;
1397 1685
1398 1686     rdb = .fdb [fdb$l_currec];             ! Get address of RDB of record to output
1399 1687
1400 1688     linedesc [dsc$u_length] = 0;           ! Init the output descriptor
1401 1689     linedesc [dsc$a_pointer] = .dif$gl_outbuf;
1402 1690     charptr = .dif$gl_outbuf;             ! Init the pointer into the output buffer
1403 1691
1404 1692     IF .cbarflag NEQ 0                     ! If change bar format output
1405 1693
1406 1694     :
1407 1695     Change bar format.  If line number is requested, then insert number
1408 1696     into the output buffer.  Then insert change bar or blanks, as
1409 1697     appropriate.
1410 1698
1411 1699     THEN BEGIN
1412 1700         IF .fdb [fdb$u_linenum]           ! If line number should be included
1413 1701         THEN BEGIN                       ! Then do so
1414 1702             insert_linenum (.rdb, linedesc, 0);
1415 1703             charptr = .charptr + dif$e_linenum;
1416 1704             END;
```

```
1417 1705
1418 1706      IF .cbarflag EQL 1
1419 1707          THEN CH$UCHAR_A (.fdb [fdb$b_cbarchr], charptr)
1420 1708          ELSE CH$UCHAR_A (XC' ', charptr);
1421 1709
1422 1710      CH$UCHAR_A (XC' ', charptr);
1423 1711
1424 1712      END
1425 1713
1426 1714      ...
1427 1715      If not changebar, then if line numbers requested then insert the line number.
1428 1716
1429 1717      ELSE IF .dif$gl_flags [dif$u_linenum]
1430 1718          THEN BEGIN
1431 1719          insert_linenum (.rdb, linedesc, 0);
1432 1720          charptr = .charptr + dif$u_linenum;
1433 1721          CH$UCHAR_A (XC' ', charptr);
1434 1722          CH$UCHAR_A (XC' ', charptr);
1435 1723          END;
1436 1724
1437 1725      ...
1438 1726      If SLP output, then handle special operators in first column.
1439 1727
1440 1728      IF .dif$gl_flags [dif$u_slp]
1441 1729          THEN IF (.rdb [rdb$w_length] GTR 0) AND
1442 1730              (CH$FIND_CH T.slpoprs [0],
1443 1731                  slpoprs [1],
1444 1732                  CH$RCHAR (rdb [rdb$t_text])) NEQ 0)
1445 1733          THEN CH$UCHAR_A (XC'<', charptr);
1446 1734
1447 1735      linedesc [dsc$w_length] = .charptr - .dif$gl_outbuf;
1448 1736      IF .dif$gl_width GTRU linedesc [dsc$w_length]
1449 1737          THEN
1450 1738          BEGIN
1451 1739          IF .dif$gl_ignore [ign$u_exact] AND .rdb [rdb$u_edited]
1452 1740          THEN
1453 1741          get_rfa_text (.fdb, linedesc, .dif$gl_width)
1454 1742          ELSE BEGIN
1455 1743          text_length = MINU (.dif$gl_width - linedesc [dsc$w_length],
1456 1744              rdb [rdb$w_length]);
1457 1745          CH$MOVE (text_length, rdb [rdb$t_text], .charptr);
1458 1746          linedesc [dsc$w_length] = linedesc [dsc$w_length] + text_length;
1459 1747          END;
1460 1748          END;
1461 1749
1462 1750      IF .dif$gl_ignore [ign$u_pretty]
1463 1751          THEN
1464 1752          linedesc [dsc$w_length] =
1465 1753          translate_tabs (.linedesc [dsc$a_pointer], .charptr, linedesc [dsc$w_length], .dif$gl_width);
1466 1754
1467 1755      put_desc (linedesc);
1468 1756
1469 1757      RETURN true;
1470 1758      END;
```

OFFC 00000 PUT_RECORD ASCII:										
		5B	000000000G	00	9E	00002	WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	1648	
		5A	000000000G	00	9E	00009	MOVAB	DIF\$GL_OUTBUF, R11		
		59	000000000G	00	9E	00010	MOVAB	DIF\$GL_IGNORE, R10		
		58	000000000V	EF	9E	00017	MOVAB	DIF\$GL_WIDTH, R9		
		5E		08	C2	0001E	MOVAB	INSERT_LINENUM, R8		
		53	04	AC	D0	00021	SUBL2	#8, SP		
		52		63	D0	00025	MOVL	FDB, R3	1686	
				6E	B4	00028	MOVL	(R3), RDB		
		50		6B	D0	0002A	CLRW	LINEDESC	1688	
	04	AE		50	D0	0002D	MOVL	DIF\$GL_OUTBUF, R0	1689	
		56		50	D0	00031	MOVL	R0, LINEDESC+4		
			08	AC	D5	00034	MOVL	R0, CHARPTR	1690	
				1E	13	00037	TSTL	CHARFLAG	1692	
				01	E1	00039	BEQL	28		
0D	24	A3		7E	D4	0003E	BBC	#1, 36(R3), 18	1700	
			04	AE	9F	00040	CLRL	-(SP)	1702	
				52	DD	00043	PUSHAB	LINEDESC		
		68		G3	FB	00045	PUSHL	RDB		
		56		06	C0	00048	CALLS	#3, INSERT_LINENUM		
		01		08	AC	0004B	ADDL2	#6, CHARPTR	1703	
					1B	12	CMPL	CHARFLAG, #1	1706	
		66		26	A3	90	BNEQ	38		
					1B	11	MOVB	38(R3), (CHARPTR)	1707	
					04	E1	BRB	48	1708	
15	000000000G	00			7E	D4	BBC	#4, DIF\$GL_FLAGS+1, 58	1717	
			04	AE	9F	00061	CLRL	-(SP)	1719	
				52	DD	00064	PUSHAB	LINEDESC		
		68		03	FB	00066	PUSHL	RDB		
		56		06	C0	00069	CALLS	#3, INSERT_LINENUM		
		66		20	90	0006C	ADDL2	#6, CHARPTR	1720	
				56	D6	0006F	MOVB	#32, (CHARPTR)	1721	
		86		20	90	00071	INCL	CHARPTR		
		20	000000000G	00	E9	00074	MOVB	#32, (CHARPTR)+	1722	
			12	A2	B5	0007B	BLBC	DIF\$GL_FLAGS+1, 78	1728	
				1B	13	0007E	TSTM	18(RDB)	1729	
		50	000000000'	EF	9A	00080	BEQL	78		
000000000'	EF	50		A2	3A	00087	MOVZBL	SLPOPRS, R0	1730	
				02	12	00090	LOCC	20(RDB), R0, SLPOPRS+1	1732	
				51	D4	00092	BNEQ	68		
				51	D5	00094	CLRL	R1		
				03	13	00096	TSTL	R1		
		86		3C	90	00098	BEQL	78		
		56		6B	A3	0009B	MOVB	#60, (CHARPTR)+	1733	
6E				69	D0	0009F	SUBW3	DIF\$GL_OUTBUF, CHARPTR, LINEDESC	1735	
		50		00	ED	000A2	MOVL	DIF\$GL_WIDTH, R0	1736	
50		6E		36	1E	000A7	CMPTV	#0, #18, LINEDESC, R0		
		15		06	E1	000A9	BGEQU	108		
		10		03	E1	000AD	BBC	#6, DIF\$GL_IGNORE, 88	1739	
	08	A2		50	DD	000B2	BBC	#3, 8(RDB), 88		
			04	AE	9F	000B4	PUSHL	R0	1741	
				53	DD	000B7	PUSHAB	LINEDESC		
				03	FB	000B9	PUSHL	R3		
				1D	11	000C0	CALLS	#3, GET_RFA_TEXT		
							BRB	108		

50	12	A2	51	6E	3C	000C2	88:	MOVZWL	LINEDESC, R1	1743
			50	51	C2	000C5		SUBL2	R1, R0	1744
			10	00	ED	000C8		CMPZV	#0, #16, 18(RDB), R0	1745
				04	1E	000CE		BGEQU	98	1746
			50	A2	3C	000D0		MOVZWL	18(RDB), R0	1747
			57	50	DD	000D4	98:	MOVL	R0, TEXT_LENGTH	1748
	66	14	A2	57	28	000D7		MOVCL	TEXT_LENGTH, 20(RDB), (CHARPTR)	1749
			6E	57	A0	000DC		ADDW2	TEXT_LENGTH, LINEDESC	1750
				6A	95	000DF	108:	TSTB	DIF\$GL_IGNORE	1751
				15	18	000E1		BGEQ	118	1752
			7E	69	DD	000E3		PUSHL	DIF\$GL_WIDTH	1753
				AE	3C	000E5		MOVZWL	LINEDESC, -(SP)	
				56	DD	000E9		PUSHL	CHARPTR	
				AE	DD	000EB		PUSHL	LINEDESC+4	
			00000000V	EF	04	FB	000EE	CALLS	#4, TRANSLATE_TABS	
			6E	50	B0	000F5		MOVW	R0, LINEDESC	
				5E	DD	000F8	118:	PUSHL	SP	1755
			00000000V	EF	01	FB	000FA	CALLS	#1, PUT_DESC	1757
			50	01	DD	00101		MOVL	#1, R0	1758
				04	00	104		RET		

; Routine Size: 261 bytes, Routine Base: \$CODE\$ + 0B26

```
1472 1759 1 ROUTINE put_record_hex_octal (fdb, cbarflag) =
1473 1760 2 BEGIN
1474 1761 3
1475 1762 4 **
1476 1763 5
1477 1764 6 FUNCTIONAL DESCRIPTION:
1478 1765 7
1479 1766 8     Format and put a record to the output file in either hex or octal.
1480 1767 9
1481 1768 10 INPUTS:
1482 1769 11
1483 1770 12     fdb =       The address of the FDB pointing to the CURREC that is
1484 1771 13             to be output.
1485 1772 14
1486 1773 15     cbarflag =  A flag that is 0 if not changebar format
1487 1774 16             1 if changebar format and bar should be output
1488 1775 17             2 if changebar format and bar should not be ouput
1489 1776 18
1490 1777 19 OUTPUTS:
1491 1778 20
1492 1779 21     The CURREC is output.
1493 1780 22
1494 1781 23 ROUTINE VALUES:
1495 1782 24
1496 1783 25     Always true
1497 1784 26
1498 1785 27 --
1499 1786 28
1500 1787 29 MAP
1501 1788 30     fdb : REF BBLOCK;
1502 1789 31
1503 1790 32 LITERAL
1504 1791 33     byte_bits = 8,           ! number of bits in a byte
1505 1792 34     hex_bits = 4,           ! number of bits in a hexadecimal digit
1506 1793 35     oct_bits = 3;           ! number of bits in an octal digit
1507 1794 36
1508 1795 37 LOCAL
1509 1796 38     additional,
1510 1797 39     bufferpointer,
1511 1798 40     bytenumber,
1512 1799 41     bytesperline,
1513 1800 42     entsinrec,
1514 1801 43     faopointer,
1515 1802 44     linedesc : BBLOCK [dsc$c_s_bln],
1516 1803 45     outputdesc : BBLOCK [dsc$c_s_bln],
1517 1804 46     padbytes,
1518 1805 47     tempfaobuf : BBLOCK [dif$c_maxfaosiz],
1519 1806 48     tempfaodesc : BBLOCK [dsc$c_s_bln],
1520 1807 49     rdb : REF BBLOCK;
1521 1808 50
1522 1809 51     rdb = .fdb [fdb$l_currec];           ! Get address of RDB of output record
1523 1810 52
1524 1811 53     !
1525 1812 54     ! Get exact or edited text, as appropriate.
1526 1813 55
1527 1814 56     IF .dif$l_ignore [ign$v_exact] AND .rdb [rdb$v_edited]
1528 1815 57     THEN BEGIN
```

```
1529 1816 linedesc [dsc$w_length] = 0;
1530 1817 linedesc [dsc$a_pointer] = .dif$gl_inbuf;
1531 1818 get_rfa_text (.fdb, linedesc,
1532 1819 MAXU (.dif$gl_masrab [rab$w_usz], .dif$gl_revraab [rab$w_usz]));
1533 1820 END
1534 1821 ELSE BEGIN
1535 1822 linedesc [dsc$w_length] = .rdb [rdb$w_length];
1536 1823 linedesc [dsc$a_pointer] = rdb [rdb$a_text];
1537 1824 END;
1538 1825
1539 1826
1540 1827 Output the record header.
1541 1828
1542 1829 put_blank ();
1543 1830 put_hex_octal_header (.rdb [rdb$a_number], .linedesc [dsc$w_length], .cbarflag);
1544 1831 put_blank ();
1545 1832
1546 1833
1547 1834 Initialize output format parameters.
1548 1835
1549 1836 bytenumber = 0;
1550 1837 bytesperline = .dif$gl_entsperline * dif$c_entsize;
1551 1838 entsinrec = (.linedesc [dsc$w_length] + dif$c_entsize - 1) / dif$c_entsize;
1552 1839 faopointer = dif$gl_faofulldesc;
1553 1840
1554 1841
1555 1842 Initialize output descriptor.
1556 1843
1557 1844 outputdesc [dsc$a_pointer] = .dif$gl_outbuf;
1558 1845 outputdesc [dsc$w_length] = .dif$gl_dumpwidth;
1559 1846
1560 1847
1561 1848 Output all the data in the record.
1562 1849
1563 1850 WHILE .entsinrec GTR 0
1564 1851 DO BEGIN
1565 1852
1566 1853
1567 1854 If less than a full line of data left, then prepare to output a partial line.
1568 1855
1569 1856 IF .linedesc [dsc$w_length] LSSU .bytesperline
1570 1857 THEN BEGIN
1571 1858 CHSCOPY (.linedesc [dsc$w_length], .linedesc [dsc$a_pointer], ! Copy partial line, zero fi
1572 1859 0, .bytesperline, .dif$gl_inbuf);
1573 1860 tempfaodesc [dsc$w_length] = dif$c_maxfaosiz; ! Set up temporary work area
1574 1861 tempfaodesc [dsc$a_pointer] = tempfaobuf;
1575 1862 SYSSFAO (dif$gl_faopartdesc, tempfaodesc, tempfaodesc, .entsinrec); ! Use FAO to build a fao con
1576 1863 faopointer = tempfaodesc; ! Use this fao string up ahe
1577 1864 bufferpointer = .dif$gl_inbuf; ! Use this data
1578 1865 END
1579 1866 ELSE bufferpointer = .linedesc [dsc$a_pointer]; ! If full line, use this dat
1580 1867
1581 1868
1582 1869 Format the output line.
1583 1870
1584 1871 dif$format_hex_octal (.bufferpointer, .dif$gl_entsperline, dif$c_entsize,
1585 1872 .bytenumber, .entsinrec, 0, .faopointer, outputdesc);
```

: Rg

1759

36	00000000G	52	04	BC	D0	000006	MOVL	2FDB, RDB	1809
31	08	00		06	E1	00000A	BBC	#6, DIF\$GL_IGNORE, 28	1814
		A2		03	E1	000012	BBC	#3, 8(RDB), 28	
	44	AE	00000000G	40	B4	000017	CLRW	LINEDESC	1816
		7E	00000000G	00	D0	00001A	MOVL	DIF\$GL_INBUF, LINEDESC+4	1817
		6E	00000000G	00	3C	000022	MOVZWL	DIF\$GL_MASRAB+32, -(SP)	1819
				00	B1	000029	CMPL	DIF\$GL_REVRAB+32, (SP)	
		6E	00000000G	00	1B	000030	BLEU	18	
				07	3C	000032	MOVZWL	DIF\$GL_REVRAB+32, (SP)	
		44		AE	9F	000039	PUSHAB	LINEDESC	1818
		04		AC	DD	00003C	PUSHL	FDB	
00000000V	EF			03	FB	00003F	CALLS	#3, GET_RFA_TEXT	
	40	AE	12	0A	11	000046	BRB	38	1814
	44	AE	14	A2	B0	000048	MOVW	18(RDB), LINEDESC	1822
00000000V	EF			A2	9E	00004D	MOVAB	20(R2), LINEDESC+4	1823
				00	FB	000052	CALLS	#0, PUT_BLANK	1829
		7E	08	AC	DD	000059	PUSHL	CBARFLAG	1830
			44	AE	3C	00005C	MOVZWL	LINEDESC, -(SP)	
		04		A2	DD	000060	PUSHL	4(RDB)	
00000000V	EF			03	FB	000063	CALLS	#3, PUT_HEX_OCTAL_HEADER	
00000000V	EF			00	FB	00006A	CALLS	#0, PUT_BLANK	1831
				6E	D4	000071	CLRL	BYTENUMBER	1836
59	00000000G	00		02	78	000073	ASHL	#2, DIF\$GL_ENTSPERLINE, BYTESPERLINE	1837
		50	40	AE	3C	00007B	MOVZWL	LINEDESC, R0	1838
		50		03	C0	00007F	ADDL2	#3, R0	
SA		50		04	C7	000082	DIVL3	#4, R0, ENTSINREC	
		5B	00000000G	00	9E	000086	MOVAB	DIF\$GL_FAOFULLDESC, FAOPOINTER	1839
	3C	AE	00000000G	00	D0	00008D	MOVL	DIF\$GL_OUTBUF, OUTPUTDESC+4	1844
	38	AE	00000000G	00	B0	00C95	MOVW	DIF\$GL_DUMPWIDTH, OUTPUTDESC	1845
				5A	D5	00009D	TSTL	ENTSINREC	1850
				03	14	00009F	BGTR	58	
				00EC	31	0000A1	BRW	118	
59	40	AE	10	58	D4	0000A4	CLRL	R8	1856
				00	ED	0000AE	CMPLV	#0, #16, LINEDESC, BYTESPERLINE	
				3D	1E	0000AC	BGEQU	68	
				58	D6	0000AE	INCL	R8	
59	00	44	00000000G	00	D0	0000B0	MOVL	DIF\$GL_INBUF, R0	1859
		BE	40	AE	2C	0000B7	MOVCS	LINEDESC, @LINEDESC+4, #0, BYTESPERLINE, -	
				60		0000BE		(R0)	
	08	AE		28	B0	0000BF	MOVW	#40, TEMPFAODESC	1860
	0C	AE	10	AE	9E	0000C3	MOVAB	TEMPFAOBUF, TEMPFAODESC+4	1861
				5A	DD	0000C8	PUSHL	ENTSINREC	1862
			0C	AE	9F	0000CA	PUSHAB	TEMPFAODESC	
			10	AE	9F	0000CD	PUSHAB	TEMPFAODESC	
			00000000G	00	9F	0000D0	PUSHAB	DIF\$GL_FAOPARTDESC	
00000000G	00			04	FB	0000D6	CALLS	#4, SYSSFAO	
	5B		08	AE	9E	0000DD	MOVAB	TEMPFAODESC, FAOPOINTER	1863
	04	AE	00000000G	00	D0	0000E1	MOVL	DIF\$GL_INBUF, BUFFERPOINTER	1864
				05	11	0000E9	BRB	78	1856
	04	AE	44	AE	D0	0000EB	MOVL	LINEDESC+4, BUFFERPOINTER	1866
			38	AE	9F	0000F0	PUSHAB	OUTPUTDESC	1871
				5B	DD	0000F3	PUSHL	FAOPOINTER	1872
				7E	D4	0000F5	CLRL	-(SP)	1871
				5A	DD	0000F7	PUSHL	ENTSINREC	1872
			10	AE	DD	0000F9	PUSHL	BYTENUMBER	
				04	DD	0000FC	PUSHL	#4	1871
			00000000G	00	DD	0000FE	PUSHL	DIF\$GL_ENTSPERLINE	

			20	AE	DD	00104	PUSHL	BUFFERPOINTER		
	00000000G	00		08	FB	00107	CALLS	#8, DIFSFORMAT_HEX_OCTAL		
		60		58	E9	0010E	BLBC	R8, 108	1878	
		57	00000000G	00	DO	00111	MOVL	DIFSGL_DUMPWIDTH, R8	1880	
		58		AE	3C	00118	MOVZWL	OUTPUTDESC, PADBYTES		
57		58	38	57	C3	0011C	SUBL3	PADBYTES, R8, PADBYTES		
		56		AE	3C	00120	MOVZWL	LINEDESC, ADDITIONAL	1886	
7E		56	40	01	7A	00124	EMUL	#1, ADDITIONAL, #0, -(SP)		
56		56		04	7B	00129	EDIV	#4, (SP)+, ADDITIONAL, ADDITIONAL		
				56	D5	0012E	TSTL	ADDITIONAL		
				22	15	00130	BLEQ	98		
50		56		03	78	00132	ASHL	#3, ADDITIONAL, R0	1892	
0C	00000000G	00		01	E1	00136	BBC	#1, DIFSGL_FLAGS, 88	1888	
		50		03	C0	0013E	ADDL2	#3, R0	1892	
		50		04	C6	00141	DIVL2	#4, R0		
56		09		50	C3	00144	SUBL3	R0, #9, ADDITIONAL		
				0A	11	00148	BRB	98	1890	
		50		02	C0	0014A	ADDL2	#2, R0	1896	
		50		03	C6	0014D	DIVL2	#3, R0		
56		0C		50	C3	00150	SUBL3	R0, #12, ADDITIONAL		
		57		56	C0	00154	ADDL2	ADDITIONAL, PADBYTES	1897	
		50		AE	3C	00157	MOVZWL	OUTPUTDESC, R0	1898	
		50	38	56	C2	0015B	SUBL2	ADDITIONAL, R0		
3C BE47		3C BE46		50	28	0015E	MOVC3	R0, @OUTPUTDESC+4[ADDITIONAL], -	1900	
								@OUTPUTDESC+4[PADBYTES]		
57		20		00	2C	00166	MOVC5	#0, (SP), #32, PADBYTES, @OUTPUTDESC+4	1901	
			3C	BE		00168				
		38	AE	58	B0	0016D	MOVW	R8, OUTPUTDESC	1902	
			38	AE	9F	00171	PUSHAB	OUTPUTDESC	1908	
00000000V		EF		01	FB	00174	CALLS	#1, PUT_DESC		
		5A	00000000G	00	C2	0017B	SUBL2	DIFSGL_ENTSPERLINE, ENTSINREC	1913	
		6E		59	C0	00182	ADDL2	BYTESPERLINE, BYTENUMBER	1914	
40		AE		59	A2	00185	SUBW2	BYTESPERLINE, LINEDESC	1915	
44		AE		59	C0	00189	ADDL2	BYTESPERLINE, LINEDESC+4	1916	
				FF0D	31	0018D	BRW	48	1850	
		50		01	DO	00190	MOVL	#1, R0	1920	
				04	00193		RET		1921	

; Routine Size: 404 bytes, Routine Base: \$CODE\$ + 0C2B

```
1636 1922 1 ROUTINE put_record_parallel (masrdb, revrdb) =
1637 1923 BEGIN
1638 1924
1639 1925 **
1640 1926
1641 1927 FUNCTIONAL DESCRIPTION:
1642 1928
1643 1929 Format and put a record to the output file in parallel mode.
1644 1930
1645 1931 INPUTS:
1646 1932
1647 1933 masrdb = The address of the RDB for the master file record.
1648 1934
1649 1935 revrdb = The address of the RDB for the revision file record.
1650 1936
1651 1937 OUTPUTS:
1652 1938
1653 1939 The specified records are output in parallel mode.
1654 1940
1655 1941 ROUTINE VALUES:
1656 1942
1657 1943 Always true
1658 1944
1659 1945 --
1660 1946
1661 1947 MAP
1662 1948 masrdb : REF BBLOCK,
1663 1949 revrdb : REF BBLOCK;
1664 1950
1665 1951 LOCAL
1666 1952 halfline, ! Amount of space for each file
1667 1953 linedesc : BBLOCK [dsc$g_s_bln], ! Descriptor for output string
1668 1954 text_length; ! Amount of space left for record text
1669 1955
1670 1956 halfline = (.dif$gl_parwidth - 5) / 2; ! Calculate amount of space for each file
1671 1957
1672 1958 linedesc [dsc$a_pointer] = .dif$gl_outbuf; ! Clear the output descriptor/buffer
1673 1959 CH$FILL ('ASCII', .dif$gl_parwidth, linedesc [dsc$a_pointer]);
1674 1960
1675 1961
1676 1962 ! If masrdb is not zero, then insert text for master file.
1677 1963 Use exact or edited text, as appropriate.
1678 1964
1679 1965 IF .masrdb NEQ 0
1680 1966 THEN IF .dif$gl_ignore [ign$u_exact] AND .masrdb [rdb$u_edited]
1681 1967 THEN BEGIN
1682 1968 linedesc [dsc$u_length] = 0;
1683 1969 dif$gl_masfdb [fdb$u_currec] = .masrdb;
1684 1970 get_rfa_text (dif$gl_masfdb, linedesc, .halfline);
1685 1971 END
1686 1972 ELSE BEGIN
1687 1973 text_length = MINU (.halfline, .masrdb [rdb$u_length]);
1688 1974 CH$MOVE (.text_length, masrdb [rdb$u_text],
1689 1975 .linedesc [dsc$a_pointer]);
1690 1976 END;
1691 1977
1692 1978 ! Remove tabs from text and insert mid-line bar.
```

```
1693 1979 2
1694 1980 3
1695 1981 4
1696 1982 5
1697 1983 6
1698 1984 7
1699 1985 8
1700 1986 9
1701 1987 10
1702 1988 11
1703 1989 12
1704 1990 13
1705 1991 14
1706 1992 15
1707 1993 16
1708 1994 17
1709 1995 18
1710 1996 19
1711 1997 20
1712 1998 21
1713 1999 22
1714 2000 23
1715 2001 24
1716 2002 25
1717 2003 26
1718 2004 27
1719 2005 28
1720 2006 29
1721 2007 30
1722 2008 31

...
translate_tabs (.dif$gl_outbuf, .dif$gl_outbuf, .halfline, .halfline);
CH$WCHAR T$ASCII ' ', .dif$gl_outbuf + .halfline + 2);

...
If revrdb is not zero, then insert text for master file.
Use exact or edited text, as appropriate.

IF .revrdb NEQ 0
  THEN IF .dif$gl_ignore [ign$u_exact] AND .revrdb [rdb$u_edited]
    THEN BEGIN
      linedesc [dsc$u_length] = .halfline + 5;
      dif$gl_revfdb [fdb$u_currec] = .revrdb;
      get_rfa_text (dif$gl_revfdb, linedesc, .dif$gl_parwidth);
    END
  ELSE BEGIN
    text_length = MINU (.halfline, .revrdb [rdb$u_length]);
    CH$MOVE (.text_length, revrdb [rdb$u_text],
      .linedesc [dsc$u_pointer] + .halfline + 5);
    END;

...
Remove tabs from remainder of text and output the line.

linedesc [dsc$u_length] =
  translate_tabs (.dif$gl_outbuf, .dif$gl_outbuf + .halfline + 5, .dif$gl_parwidth, .dif$gl_parwidth);
put_desc (linedesc);

RETURN true;
END;
```

OFFC 00000 PUT_RECORD PARALLEL:

		5B	00000000G	00	9E	00002	WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	1922
		5A	00000000G	00	9E	00009	MOVAB	DIF\$GL_MASFDB, R11	
		59	00000000G	00	9E	0C010	MOVAB	DIF\$GL_IGNORE, R10	
		58	00000000G	00	9E	00017	MOVAB	DIF\$GL_PARWIDTH, R9	
		5E		08	C2	0001E	SUBL2	#8, SP	
		51		69	D0	00021	MOVL	DIF\$GL_PARWIDTH, R1	1956
	56	50	FB	A1	9E	00024	MOVAB	-5(R1), R0	
		50		02	C7	00028	DIVL3	#2, R0, HALF LINE	
51	20	AE		68	D0	0002C	MOVL	DIF\$GL_OUTBUF, LINEDESC+4	1958
		6E		00	2C	00030	MOVCS	#0, (SP), #32, R1, @LINEDESC+4	1959
		52	04	BE		00035			
			04	AC	D0	00037	MOVL	MASRDB, R2	1965
				36	13	00038	BEQL	38	
	1A	6A		06	E1	0003D	BBC	#6, DIF\$GL_IGNORE, 18	1966
	15	A2	08	03	E1	00041	BBC	#3, 8(R2), -18	
				6E	B4	00046	CLRW	LINEDESC	1968
		68		52	D0	00048	MOVL	R2, DIF\$GL_MASFDB	1969
				56	DD	0004B	PUSHL	HALF LINE	1970
			04	AE	9F	0004D	PUSHAB	LINEDESC	
				5B	DD	00050	PUSHL	R11	

50	12	A2	50	10	00000000V	EF	03	FB	00052	CALLS	#3, GET_RFA_TEXT	1966	
			50	10			18	11	00059	BRB	38	1973	
			50	10			56	DO	0005B	18:	MOVL	HALFLINE, R0	
			50	10			00	ED	0005E		CMPZV	#0, #16, 18(R2), R0	
			50	10			04	1E	00064		BGEQU	28	
			50	10			A2	3C	00066		MOVZWL	18(R2), R0	
			50	10			50	DO	0006A	28:	MOVL	R0, TEXT_LENGTH	
			50	10			57	28	0006D		MOVC3	TEXT_LENGTH, 20(R2), 2LINEDESC+4	1975
			50	10			56	DD	00073	38:	PUSHL	HALFLINE	1980
			50	10			56	DD	00075		PUSHL	HALFLINE	
			50	10			68	DO	00077		MOVL	DIF\$GL_OUTBUF, R0	
			50	10			50	DD	0007A		PUSHL	R0	
			50	10			50	DD	0007C		PUSHL	R0	
			50	10			04	FB	0007E		CALLS	#4, TRANSLATE_TABS	
			50	10			68	DO	00085		MOVL	DIF\$GL_OUTBUF, R0	1981
			50	10			8F	90	00088		MOVB	#124, 2(HALFLINE)[R0]	
			50	10			AC	DO	0008E		MOVL	REVRDB, R2	1987
			50	10			45	13	00092		BEQL	68	
			50	10			06	E1	00094		BBC	#6, DIF\$GL_IGNORE, 48	1988
			50	10			03	E1	00098		BBC	#3, 8(R2), -48	
			50	10			05	A1	0009D		ADDW3	#5, HALFLINE, LINEDESC	1990
			50	10			52	DO	000A1		MOVL	R2, DIF\$GL_REVFD	1991
			50	10			69	DD	000AB		PUSHL	DIF\$GL_PARWIDTH	1992
			50	10			AE	9F	000AA		PUSHAB	LINEDESC	
			50	10			00	9F	000AD		PUSHAB	DIF\$GL_REVFD	
			50	10			03	FB	000B3		CALLS	#3, GET_RFA_TEXT	
			50	10			1D	11	000BA		BRB	68	1988
			50	10			56	DO	000BC	48:	MOVL	HALFLINE, R0	1995
			50	10			00	ED	000BF		CMPZV	#0, #16, 18(R2), R0	
			50	10			04	1E	000C5		BGEQU	58	
			50	10			A2	3C	000C7		MOVZWL	18(R2), R0	
			50	10			50	DO	000CB	58:	MOVL	R0, TEXT_LENGTH	
			50	10			AE	C1	000CE		ADDL3	LINEDESC+4, HALFLINE, R0	1997
			50	10			57	28	000D3		MOVC3	TEXT_LENGTH, 20(R2), 5(R0)	
			50	10			69	DO	000D9	68:	MOVL	DIF\$GL_PARWIDTH, R0	2004
			50	10			50	DD	000DC		PUSHL	R0	
			50	10			50	DD	000DE		PUSHL	R0	
			50	10			68	DO	000E0		MOVL	DIF\$GL_OUTBUF, R0	
			50	10			50	9F	000E3		PUSHAB	5(HALFLINE)[R0]	
			50	10			50	DD	000E7		PUSHL	R0	
			50	10			04	FB	000E9		CALLS	#4, TRANSLATE_TABS	
			50	10			50	BO	000F0		MOVW	R0, LINEDESC	
			50	10			5E	DD	000F3		PUSHL	SP	2005
			50	10			01	FB	000F5		CALLS	#1, PUT_DESC	
			50	10			01	DO	000FC		MOVL	#1, R0	2007
			50	10			04	DO	000FF		RET		2008

; Routine Size: 256 bytes, Routine Base: \$CODE\$ + 0DBF

```
1724 2009 1 ROUTINE translate_tabs (bufaddr, start, buflen, maxlen) =
1725 2010 BEGIN
1726 2011
1727 2012 ++
1728 2013
1729 2014 FUNCTIONAL DESCRIPTION:
1730 2015
1731 2016 Convert the tabs in the buffer, starting at start, to spaces.
1732 2017 Also replace line feeds, form feeds, and carriage returns with
1733 2018 appropriate text.
1734 2019
1735 2020 INPUTS:
1736 2021
1737 2022 bufaddr = Address of the input buffer.
1738 2023
1739 2024 start = Address of first char to examine.
1740 2025
1741 2026 buflen = Length of the input text.
1742 2027
1743 2028 maxlen = Length of buffer
1744 2029
1745 2030 OUTPUTS:
1746 2031
1747 2032 All tabs are converted to spaces.
1748 2033
1749 2034 ROUTINE VALUES:
1750 2035
1751 2036 Length of translated text
1752 2037
1753 2038 --
1754 2039 LOCAL
1755 2040 charptr, ! Address of character to translate
1756 2041 endptr, ! Address of last byte to translate+1
1757 2042 maxptr, ! Address of last byte of buffer+1
1758 2043 fill, ! Number of characters to insert
1759 2044 textaddr : REF VECTOR [, BYTE]; ! Address of text to insert
1760 2045
1761 2046 charptr = .start; ! Starting character
1762 2047 endptr = .bufaddr + .buflen; ! Last character to translate+1
1763 2048 maxptr = .bufaddr + .maxlen; ! End of buffer+1
1764 2049
1765 2050 WHILE (.charptr LSSA .endptr) ! Examine all characters
1766 2051 DO BEGIN
1767 2052 IF (CHSRCHAR (.charptr) GEQU 'X'20'
1768 2053 AND (CHSRCHAR (.charptr) LEQU 'X'7E'
1769 2054 THEN ! Printing character
1770 2055 charptr = .charptr + 1
1771 2056 ELSE
1772 2057 BEGIN
1773 2058 SELECTONE (CHSRCHAR (.charptr) OF
1774 2059 SET
1775 2060 ['X'09']): ! Tab
1776 2061 BEGIN
1777 2062 fill = 8 - ((.charptr - .start) MOD 8); ! Calculate number of spaces to use
1778 2063 textaddr = blanks [fill]; ! Insert blanks
1779 2064 END;
1780 2065
```

```
1781 2066 4 [X'0A']: ! Line feed
1782 2067 5 BEGIN
1783 2068 6 textaddr = lf; ! Insert <LF>
1784 2069 7 fill = .textaddr [0]; ! Length of string
1785 2070 8 textaddr = textaddr [1]; ! Text address
1786 2071 9 END;
1787 2072 4
1788 2073 5 [X'0B']: ! Vertical tab
1789 2074 6 BEGIN
1790 2075 7 textaddr = vt; ! Insert <CR>
1791 2076 8 fill = .textaddr [0]; ! Length of string
1792 2077 9 textaddr = textaddr [1]; ! Text address
1793 2078 4
1794 2079 5 [X'0C']: ! Form feed
1795 2080 6 BEGIN
1796 2081 7 textaddr = ff; ! Insert <FF>
1797 2082 8 fill = .textaddr [0]; ! Length of string
1798 2083 9 textaddr = textaddr [1]; ! Text address
1800 2085 4
1801 2086 5 [X'0D']: ! Carriage return
1802 2087 6 BEGIN
1803 2088 7 textaddr = cr; ! Insert <CR>
1804 2089 8 fill = .textaddr [0]; ! Length of string
1805 2090 9 textaddr = textaddr [1]; ! Text address
1806 2091 4
1807 2092 5 [OTHERWISE]: ! All other characters
1808 2093 6 BEGIN
1809 2094 7 textaddr = period; ! Insert period
1810 2095 8 fill = .textaddr [0]; ! Length of string
1811 2096 9 textaddr = textaddr [1]; ! Text address
1812 2097 4
1813 2098 5
1814 2099 6
1815 2100 7
1816 2101 8
1817 2102 9
1818 2103 4
1819 2104 5 IF .fill GTR .maxptr - .charptr ! Shorten it if it goes past end of buffer
1820 2105 6 THEN fill = .maxptr - .charptr;
1821 2106 7 endptr = CHSMOVE ( ! Shift the unexamined text over
1822 2107 8 MIN ( ! and update endptr
1823 2108 9 .maxptr - .charptr - .fill, ! Target space available
1824 2109 4 .endptr - .charptr - 1), ! Number of source characters remaining
1825 2110 5 .charptr+1,
1826 2111 6 .charptr + .fill);
1827 2112 7 charptr = CHSMOVE (.fill, .textaddr, .charptr); ! Insert the text and update charptr
1828 2113 8 END;
1829 2114 9
1830 2115 4
1831 2116 5 END;
RETURN .charptr-.bufaddr;
END;
```

OFFC 00000 TRANSLATE_TABS:

		5B	00000000	EF	9E	00002	WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	2009
		58	08	AC	D0	00009	MOVAB	BLANKS+1, R11	
5A	04	AC	0C	AC	C1	0000D	MOVL	START, CHARPTR	2046
57	04	AC	10	AC	C1	00013	ADDL3	BUFLN, BUFADDR, ENDPTR	2047
		5A		58	D1	00019	ADDL3	MAXLN, BUFADDR, MAXPTR	2048
				03	1F	0001C	CMPB	CHARPTR, ENDPTR	2050
				008F	31	0001E	BLSSU	28	
		20		68	91	00021	BRW	138	
				0A	1F	00024	CMPB	(CHARPTR), #32	2052
7E	8F			68	91	00026	BLSSU	38	
				04	1A	0002A	CMPB	(CHARPTR), #126	2053
				58	D6	0002C	BGTRU	38	
				E9	11	0002E	INCL	CHARPTR	2055
		09		68	91	00030	BRB	18	
				18	12	00033	CMPB	(CHARPTR), #9	2060
				58	C3	00035	BNEQ	48	
50	08	AC		01	7A	0003A	SUBL3	CHARPTR, START, R0	2062
00		50		08	7B	0003F	EMUL	#1, R0, #0, -(SP)	
50		8E		A0	9E	00044	EDIV	#8, (SP)+, R0, R0	
		56	08	68	9E	00048	MOVAB	8(R0), FILL	
		59		33	11	0004B	MOVAB	BLANKS+1, TEXTADDR	2063
				68	91	0004D	BRB	108	2058
		0A		06	12	00050	CMPB	(CHARPTR), #10	2066
		59	EF	AB	9E	00052	BNEQ	58	
				25	11	00056	MOVAB	LF, TEXTADDR	2068
		08		68	91	00058	BRB	98	2069
				06	12	0005B	CMPB	(CHARPTR), #11	2073
		59	F7	AB	9E	0005D	BNEQ	68	
				1A	11	00061	MOVAB	VT, TEXTADDR	2075
		0C		68	91	00063	BRB	98	2076
				06	12	00066	CMPB	(CHARPTR), #12	2080
		59	E7	AB	9E	00068	BNEQ	78	
				0F	11	0006C	MOVAB	FF, TEXTADDR	2082
		0D		68	91	0006E	BRB	98	2083
				06	12	00071	CMPB	(CHARPTR), #13	2087
		59	DF	AB	9E	00073	BNEQ	88	
				04	11	00077	MOVAB	CR, TEXTADDR	2089
		59	0B	AB	9E	00079	BRB	98	2090
		56		89	9A	0007D	MOVAB	PERIOD, TEXTADDR	2096
50		57		58	C3	00080	(TEXTADDR)+, FILL		2097
		50		56	D1	00084	CHARPTR, MAXPTR, R0		2097
				03	15	00087	CMPB	FILL, R0	2103
		56		50	D0	00089	BLEQ	118	
		50		56	C2	0008C	MOVL	R0, FILL	2104
53		5A		58	C3	0008F	SUBL2	FILL, R0	2107
				53	D7	00093	SUBL3	CHARPTR, ENDPTR, R3	2108
		53		50	D1	00095	DECL	R3	
				03	15	00098	CMPB	R0, R3	
		50		53	D0	0009A	BLEQ	128	
6648	01	AB		50	28	0009D	MOVL	R3, R0	
		5A		53	D0	000A3	MOVAB	R0, 1(CHARPTR), (FILL)[CHARPTR]	2110
68		69		56	28	000A6	MOVL	R3, ENDPTR	
		58		53	D0	000AA	MOVAB	FILL, (TEXTADDR), (CHARPTR)	2111
				FF69	31	000AD	MOVL	R3, CHARPTR	
53		58	04	AC	C3	000B0	BRW	18	2050
		50		53	D0	000B5	SUBL3	BUFADDR, CHARPTR, R3	2115
							MOVL	R3, R0	

DIF
VO4

: 2116 :

; Routine Size: 185 bytes. Routine Base: SCODES + 0EBF

```
1833 2117 1 ROUTINE put_hex_octal_header (number, length, cbarflag) =
1834 2118 BEGIN
1835 2119
1836 2120
1837 2121
1838 2122
1839 2123
1840 2124
1841 2125
1842 2126
1843 2127
1844 2128
1845 2129
1846 2130
1847 2131
1848 2132
1849 2133
1850 2134
1851 2135
1852 2136
1853 2137
1854 2138
1855 2139
1856 2140
1857 2141
1858 2142
1859 2143
1860 2144
1861 2145
1862 2146
1863 2147
1864 2148
1865 2149
1866 2150
1867 2151
1868 2152
1869 2153
1870 2154
1871 2155
1872 2156
1873 2157
1874 2158
1875 2159
1876 2160
1877 2161
1878 2162
1879 2163
1880 2164
1881 2165
1882 2166
1883 2167
1884 2168
1885 2169
1886 2170
1887 2171
1888 2172
1889 2173

ROUTINE put_hex_octal_header (number, length, cbarflag) =
BEGIN
--
FUNCTIONAL DESCRIPTION:
    Put a hex or octal record header to the output file.
INPUTS:
    number = The number of the record.
    length = The length of the record in bytes.
    cbarflag = A flag that is 0 if not changebar format
               1 if changebar format and bar should be output
               2 if changebar format and bar should not be ouput
OUTPUTS:
    None
ROUTINE VALUES:
    Always true
--
LOCAL
    changedesc : BBLOCK [dsc$c_s_bln],
    faodesc : BBLOCK [dsc$c_s_bln],
    linedesc : BBLOCK [dsc$c_s_bln];
    linedesc [dsc$w_length] = .dif$gl_dumpwidth;
    linedesc [dsc$a_pointer] = .dif$gl_outbuf;
    IF .dif$gl_flags [dif$v_hex]
    THEN BEGIN
        faodesc [dsc$w_length] = .hexheader [0];
        faodesc [dsc$a_pointer] = hexheader [1];
    END
    ELSE BEGIN
        faodesc [dsc$w_length] = .octheader [0];
        faodesc [dsc$a_pointer] = octheader [1];
    END;
    changedesc [dsc$a_pointer] = change [1];
    IF .cbarflag
    THEN changedesc [dsc$w_length] = .change [0]
    ELSE changedesc [dsc$w_length] = 0;
    Generate output string and write it to the file.
SYSSFAO (faodesc, linedesc, linedesc, .number, .number, .length, .length, changedesc);
put_desc (linedesc);
```

```
: 1890
: 1891
: 1892
```

```
2174 2
2175 2 RETURN true;
2176 1 END;
```

```
0004 00000 PUT_HEX_OCTAL_HEADER:
      52 00000000' EF 9E 00002 .WORD Save R2
      5E 18 C2 00009 MOVAB HEXHEADER, R2
      6E 00000000G 00 B0 0000C SUBL2 #24, SP
      AE 00000000G 00 D0 00013 MOVW DIF$GL_DUMPWIDTH, LINEDESC
0B 00000000G 00 01 E1 0001B MOVL DIF$GL_OUTBUF, LINEDESC+4
      08 AE 62 9B 00023 BBC #1, DIF$GL_FLAGS, 1$
      0C AE 01 A2 9E 00027 MOVZBW HEXHEADER, FAODESC
      08 AE 6C A2 9B 0002E 1$: MOVZBW OCTHEADER, FAODESC
      0C AE 6D A2 9E 00033 MOVAB OCTHEADER+1, FAODESC+4
      14 AE FF79 C2 9E 00038 2$: MOVAB CHANGE+1, CHANGEDESC+4
      08 AE 0C AC E9 0003E BLBC C$ARFLAG, 3$
      10 AE FF78 C2 9B 00042 MOVZBW CHANGE, CHANGEDESC
      10 AE 03 11 00048 BRB 4$
      10 AE B4 0004A 3$: CLRW CHANGEDESC
      08 AE 9F 0004D 4$: PUSHAB CHANGEDESC
      7E 04 AC DD 00050 PUSHL LENGTH
      04 AC 7D 00053 MOVQ NUMBER, -(SP)
      14 AE 9F 0005A PUSHL NUMBER
      18 AE 9F 0005D PUSHAB LINEDESC
      24 AE 9F 00060 PUSHAB LINEDESC
00000000G 00 08 FB 00063 PUSHAB FAODESC
00000000V EF 5E DD 0006A CALLS #8, SYSSFAO
      50 01 FB 0006C PUSHL SP
      01 D0 00073 CALLS #1, PUT_DESC
      04 00076 MOVL #1, R0
      RET
```

```
2117
2151
2152
2154
2156
2157
2154
2160
2161
2164
2165
2166
2167
2172
2173
2175
2176
```

; Routine Size: 119 bytes, Routine Base: \$CODE\$ + 0F78

```
1894 2177 1 ROUTINE put_idline (fdb) =
1895 2178 BEGIN
1896 2179
1897 2180 ++
1898 2181
1899 2182 FUNCTIONAL DESCRIPTION:
1900 2183
1901 2184 Put a record containing file id line to the output file.
1902 2185
1903 2186 INPUTS:
1904 2187
1905 2188 fdb = The address of the FDB of the file to be described.
1906 2189
1907 2190 OUTPUTS:
1908 2191
1909 2192 None
1910 2193
1911 2194 ROUTINE VALUES:
1912 2195
1913 2196 Always true
1914 2197
1915 2198 --
1916 2199
1917 2200 MAP
1918 2201 fdb : REF BBLOCK;
1919 2202
1920 2203 LOCAL
1921 2204 linedesc : BBLOCK [dsc$c_s_bln]; ! Descriptor for the output string
1922 2205
1923 2206 BIND
1924 2207 filedesc = fdb [fdb$l_filedesc] : REF BBLOCK [dsc$c_s_bln]; ! resultant file name string descriptor
1925 2208
1926 2209 linedesc [dsc$a_pointer] = .dif$gl_outbuf; ! Init output desc pointer to buffer
1927 2210 linedesc [dsc$w_length] = MINU ( ! Set output desc size
1928 2211 .file [0] + .filedesc [dsc$w_length],
1929 2212 .dif$gl_width);
1930 2213
1931 2214 CH$COPY (
1932 2215 .file [0], file [1], ! 'FILE' string
1933 2216 .filedesc [dsc$w_length], .filedesc [dsc$a_pointer], ! resultant file name
1934 2217 0, ! fill never used
1935 2218 .linedesc [dsc$w_length], .linedesc [dsc$a_pointer]); ! line buffer
1936 2219
1937 2220 put_desc (linedesc); ! Output id line
1938 2221
1939 2222 RETURN true;
1940 2223 END;
```

```
OFFC 00000 PUT_IDLINE:
          SB 00000000' EF 9E 00002 .WORD Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11 : 2177
          SE 08 C2 00009 MOVAB FILE, R11
          AC 38 C1 0000C SUBL2 #8, SP : 2207
          50 04 ADDL3 #56, FDB, R0
```


0 10
15-Sep-1984 23:43:35 VAX-11 Bliss-32 V4.0-742 Page 70
14-Sep-1984 12:19:23 DISK\$VMSMASTER:[DIF.SRC]OUTPUT.B32;1 (21)

Address	Hex	Op	Op2	Op3	Op4	Op5	Op6	Op7	Op8	Op9	Op10	Op11	Op12	Op13	Op14	Op15	Op16	Op17	Op18	Op19	Op20	Op21	Op22	Op23	Op24	Op25	Op26	Op27	Op28	Op29	Op30	Op31	Op32	Op33	Op34	Op35	Op36	Op37	Op38	Op39	Op40	Op41	Op42	Op43	Op44	Op45	Op46	Op47	Op48	Op49	Op50	Op51	Op52	Op53	Op54	Op55	Op56	Op57	Op58	Op59	Op60	Op61	Op62	Op63	Op64	Op65	Op66	Op67	Op68	Op69	Op70	Op71	Op72	Op73	Op74	Op75	Op76	Op77	Op78	Op79	Op80	Op81	Op82	Op83	Op84	Op85	Op86	Op87	Op88	Op89	Op90	Op91	Op92	Op93	Op94	Op95	Op96	Op97	Op98	Op99	Op100	Op101	Op102	Op103	Op104	Op105	Op106	Op107	Op108	Op109	Op110	Op111	Op112	Op113	Op114	Op115	Op116	Op117	Op118	Op119	Op120	Op121	Op122	Op123	Op124	Op125	Op126	Op127	Op128	Op129	Op130	Op131	Op132	Op133	Op134	Op135	Op136	Op137	Op138	Op139	Op140	Op141	Op142	Op143	Op144	Op145	Op146	Op147	Op148	Op149	Op150	Op151	Op152	Op153	Op154	Op155	Op156	Op157	Op158	Op159	Op160	Op161	Op162	Op163	Op164	Op165	Op166	Op167	Op168	Op169	Op170	Op171	Op172	Op173	Op174	Op175	Op176	Op177	Op178	Op179	Op180	Op181	Op182	Op183	Op184	Op185	Op186	Op187	Op188	Op189	Op190	Op191	Op192	Op193	Op194	Op195	Op196	Op197	Op198	Op199	Op200	Op201	Op202	Op203	Op204	Op205	Op206	Op207	Op208	Op209	Op210	Op211	Op212	Op213	Op214	Op215	Op216	Op217	Op218	Op219	Op220	Op221	Op222	Op223	Op224	Op225	Op226	Op227	Op228	Op229	Op230	Op231	Op232	Op233	Op234	Op235	Op236	Op237	Op238	Op239	Op240	Op241	Op242	Op243	Op244	Op245	Op246	Op247	Op248	Op249	Op250	Op251	Op252	Op253	Op254	Op255	Op256	Op257	Op258	Op259	Op260	Op261	Op262	Op263	Op264	Op265	Op266	Op267	Op268	Op269	Op270	Op271	Op272	Op273	Op274	Op275	Op276	Op277	Op278	Op279	Op280	Op281	Op282	Op283	Op284	Op285	Op286	Op287	Op288	Op289	Op290	Op291	Op292	Op293	Op294	Op295	Op296	Op297	Op298	Op299	Op300	Op301	Op302	Op303	Op304	Op305	Op306	Op307	Op308	Op309	Op310	Op311	Op312	Op313	Op314	Op315	Op316	Op317	Op318	Op319	Op320	Op321	Op322	Op323	Op324	Op325	Op326	Op327	Op328	Op329	Op330	Op331	Op332	Op333	Op334	Op335	Op336	Op337	Op338	Op339	Op340	Op341	Op342	Op343	Op344	Op345	Op346	Op347	Op348	Op349	Op350	Op351	Op352	Op353	Op354	Op355	Op356	Op357	Op358	Op359	Op360	Op361	Op362	Op363	Op364	Op365	Op366	Op367	Op368	Op369	Op370	Op371	Op372	Op373	Op374	Op375	Op376	Op377	Op378	Op379	Op380	Op381	Op382	Op383	Op384	Op385	Op386	Op387	Op388	Op389	Op390	Op391	Op392	Op393	Op394	Op395	Op396	Op397	Op398	Op399	Op400	Op401	Op402	Op403	Op404	Op405	Op406	Op407	Op408	Op409	Op410	Op411	Op412	Op413	Op414	Op415	Op416	Op417	Op418
---------	-----	----	-----	-----	-----	-----	-----	-----	-----	-----	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------	-------

; Routine Size: 104 bytes, Routine Base: \$CODES + 0FEF

```
1942 2224 1 ROUTINE put_parallel_idline =
1943 2225 BEGIN
1944 2226
1945 2227 ++
1946 2228
1947 2229 FUNCTIONAL DESCRIPTION:
1948 2230
1949 2231 Put a record containing both file ids to the output file,
1950 2232 in parallel format.
1951 2233
1952 2234 INPUTS:
1953 2235
1954 2236 None
1955 2237
1956 2238 OUTPUTS:
1957 2239
1958 2240 None
1959 2241
1960 2242 ROUTINE VALUES:
1961 2243
1962 2244 Always true
1963 2245
1964 2246 --
1965 2247
1966 2248 LOCAL
1967 2249 charptr, ! Pointer into the output descriptor
1968 2250 linedesc : BBLOCK [dsc$s_bln], ! Descriptor for the output string
1969 2251 namesize; ! Space left for each file name
1970 2252
1971 2253 dif$gl_parwidth = 2 * ((dif$gl_width - 1) / 2) + 1; ! Calculate width of parallel listing
1972 2254 linedesc [dsc$w_length] = .dif$gl_parwidth; ! Init output descriptor
1973 2255 linedesc [dsc$a_pointer] = .dif$gl_outbuf;
1974 2256 namesize = (.dif$gl_parwidth - 5) / 2; ! Calculate space left for each file name
1975 2257
1976 2258
1977 2259 Output a line of dashes.
1978 2260
1979 2261 CH$FILL (XASCII '-', .linedesc [dsc$w_length], .linedesc [dsc$a_pointer]);
1980 2262 put_desc (linedesc);
1981 2263
1982 2264
1983 2265 Build line with both file names.
1984 2266
1985 2267 charptr = CH$COPY (
1986 2268 file [0], ! Insert 'FILE'
1987 2269 file [1],
1988 2270 .dif$gl_masdesc [dsc$w_length], ! Insert master file name
1989 2271 .dif$gl_masdesc [dsc$a_pointer],
1990 2272 XASCII ' ', ! blank fill
1991 2273 namesize, ! buffer length
1992 2274 .linedesc [dsc$a_pointer]); ! buffer address
1993 2275 CH$WCHAR-A (XASCII ' ', charptr);
1994 2276 CH$WCHAR-A (XASCII ' ', charptr);
1995 2277 CH$WCHAR-A (XASCII ' ', charptr); ! insert mid-line bar
1996 2278 CH$WCHAR-A (XASCII ' ', charptr);
1997 2279 CH$WCHAR-A (XASCII ' ', charptr);
1998 2280 CH$COPY T
```

```
1999 2281 2 .file [0],
2000 2282 2 .file [1],
2001 2283 2 .dif$gl_revdesc [dsc$w_length],
2002 2284 2 .dif$gl_revdesc [dsc$a_pointer],
2003 2285 2 ASCII,
2004 2286 2 .namesize,
2005 2287 2 .charptr);
2006 2288
2007 2289 put_desc (linedesc);
2008 2290
2009 2291 RETURN true;
2010 2292 1 END;
```

```
! Insert 'FILE'
! Insert revision file name
! blank fill
! buffer length
! buffer address
! Output the line
```

```
OFFC 00000 PUT_PARALLEL IDLINE:
      5E 08 C2 00002 .WORD Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11 2224
      50 01 C3 00005 SUBL2 #8, SP 2253
      00000000G 50 02 C6 0000D DIVL2 #1, DIF$GL_WIDTH, R0
      00 01 78 00010 ASHL #2, R0
      00000000G 00 00 D6 00018 INCL #1, R0, DIF$GL_PARWIDTH
      50 00 D0 0001E DIF$GL_PARWIDTH
      6E 50 B0 00025 MOVL DIF$GL_PARWIDTH, R0 2254
      04 AE 00000000G 00 00 D0 00028 MOVW R0, LINEDESC
      50 05 C2 00030 MOVL DIF$GL_OUTBUF, LINEDESC+4 2255
      56 50 02 C7 00033 SUBL2 #5, R0 2256
      2D 6E 00 00 2C 00037 DIVL3 #2, R0, NAMESIZE
      04 BE 0003C MOVC5 #0, (SP), #45, LINEDESC, @LINEDESC+4 2261
      5E DD 0003E PUSHL SP 2262
      00000000V EF 01 FB 00040 CALLS #1, PUT_DESC
      5B 00000000' EF 9A 00047 MOVZBL FILE, R11 2268
      5A 00000000G 00 3C 0004E MOVZWL DIF$GL_MASDESC, R10 2270
      59 00000000G 00 D0 00055 MOVL DIF$GL_MASDESC+4, R9 2271
      58 56 D0 0005C MOVL NAMESIZE, R8 2273
      57 04 AE D0 0005F MOVL LINEDESC+4, R7 2274
      5B 5B 2C 00063 MOVC5 R11, FILE+1, #32, R8, (R7)
      67 0006C
      0C 18 0006D BGEQ 18
      57 5B C0 0006F ADDL2 R11, R7
      58 5B C2 00072 SUBL2 R11, R8
      20 69 5A 2C 00075 MOVC5 R10, (R9), #32, R8, (R7)
      83 207C2020 67 0007A
      83 8F D0 0007B 18: MOVL #545005600, (CHARPTR)+ 2275
      5A 00000000' 20 90 00082 MOVW #32, (CHARPTR)+ 2279
      59 00000000G EF 9A 00085 MOVZBL FILE, R10 2281
      58 00000000G 00 3C 0008C MOVZWL DIF$GL_REVDESC, R9 2283
      57 00000000G 00 D0 00093 MOVL DIF$GL_REVDESC+4, R8 2284
      56 20 00000000' 53 D0 0009A MOVL CHARPTR, R7 2287
      EF 5A 2C 0009D MOVC5 R10, FILE+1, #32, R6, (R7)
      67 000A6
      0C 18 000A7 BGEQ 28
      57 5A C0 000A9 ADDL2 R10, R7
      56 5A C2 000AC SUBL2 R10, R6
      68 59 2C 000AF MOVC5 R9, (R8), #32, R6, (R7)
```

DIF_OUTPUT
V04=000

E 10
13-Sep-1984 23:43:35
14-Sep-1984 12:19:23

VAX-11 Bliss-32 V4.0-742
DISK8VMSMASTER:[DIF.SRC]OUTPUT.B32;1 Page 73
(22)

00000000v EF
50

67 000B4
5E DD 000B5 28:
01 FB 000B7
01 D0 000BE
04 000C1

PUSHL SP
CALLS #1, PUT_DESC
MOVL #1, R0
RET

: 2289
:
:
:
: 2291
:
: 2292

: Routine Size: 194 bytes, Routine Base: \$CODES + 1057


```
2012 2293 1 ROUTINE put_blank =
2013 2294 BEGIN
2014 2295
2015 2296 ++
2016 2297
2017 2298 FUNCTIONAL DESCRIPTION:
2018 2299
2019 2300 Put a blank line to the output file.
2020 2301
2021 2302 INPUTS:
2022 2303
2023 2304 None
2024 2305
2025 2306 OUTPUTS:
2026 2307
2027 2308 None
2028 2309
2029 2310 ROUTINE VALUES:
2030 2311
2031 2312 Always true
2032 2313
2033 2314 --
2034 2315
2035 2316 LOCAL
2036 2317 linedesc : BBLOCK [dsc$a_s_bln],
2037 2318 tempbuf;
2038 2319
2039 2320 linedesc [dsc$a_length] = 0;
2040 2321 linedesc [dsc$a_pointer] = tempbuf;
2041 2322
2042 2323 put_desc (linedesc);
2043 2324
2044 2325 RETURN true;
2045 2326 1 END;
```

0000 00000 PUT_BLANK:						
	5E		0C C2 00002	WORD	Save nothing	2293
		04	AE B4 00005	SUBL2	#12, SP	
	08 AE		6E 9E 00008	CLRW	LINEDESC	2320
		04	AE 9F 0000C	MOVAB	TEMPBUF, LINEDESC+4	2321
00000000V	EF		01 FB 0000F	PUSHAB	LINEDESC	2323
	50		01 D0 00016	CALLS	#1, PUT_DESC	
			04 00019	MOVL	#1, R0	2325
				RET		2326

; Routine Size: 26 bytes, Routine Base: \$CODE\$ + 1119

```
2047 2327 1 ROUTINE put_desc (linedesc) =
2048 2328 BEGIN
2049 2329
2050 2330
2051 2331
2052 2332
2053 2333
2054 2334
2055 2335
2056 2336
2057 2337
2058 2338
2059 2339
2060 2340
2061 2341
2062 2342
2063 2343
2064 2344
2065 2345
2066 2346
2067 2347
2068 2348
2069 2349
2070 2350
2071 2351
2072 2352
2073 2353
2074 2354
2075 2355
2076 2356
2077 2357
2078 2358
2079 2359
2080 2360
2081 2361
2082 2362
2083 2363
2084 2364
2085 2365
```

ROUTINE put_desc (linedesc) =
BEGIN

++

FUNCTIONAL DESCRIPTION:

Put a descriptor to the output file.

INPUTS:

linedesc = The address of the string descriptor for the text
that is to be output.

OUTPUTS:

None

ROUTINE VALUES:

Always true

--

MAP

linedesc : REF BBLOCK;

LOCAL

status;

dif\$gl_outtab [rab\$w_rsz] = .linedesc [dsc\$w_length];
dif\$gl_outtab [rab\$l_rbf] = .linedesc [dsc\$a_pointer];

IF NOT (status = \$PUT (RAB = dif\$gl_outtab))
THEN SIGNAL (dif\$writeerr, 1, dif\$gl_outdesc, .status,
.dif\$gl_outtab [rab\$l_stv]);

RETURN true;

END;

```
                                .EXTRN  SYS$PUT
                                0004 00000 PUT_DESC:
                                .WORD    Save R2
                                52 00000000G 00 9E 00002  MOVAB  DIF$GL_OUTTAB+34, R2
                                50          04 AC D0 00009  MOVL   LINEDESC, R0
                                62          60 B0 0000D  MOVW   (R0), DIF$GL_OUTTAB+34
                                06 A2          04 A0 D0 00010  MOVL   4(R0), DIF$GL_OUTTAB+40
                                DE          A2 9F 00015  PUSHAB DIF$GL_OUTTAB
                                00000000G 00 01 FB 00018  CALLS  #1, SYS$PUT
                                1A          50 E8 0001F  BLBS   STATUS, 18
                                EA          A2 DD 00022  PUSHL  DIF$GL_OUTTAB+12
                                50          DD 00025  PUSHL  STATUS
                                00000000G 00 9F 00027  PUSHAB DIF$GL_OUTDESC
                                01 DD 0002D  PUSHL  #1
```

2327
2357
2358
2360
2362
2361

DIF OUTPUT
V04=000

H 10
15-Sep-1984 23:43:35
14-Sep-1984 12:19:23

VAX-11 Bliss-32 V4.0-742
DISK&VMSMASTER:[DIF.SRC]OUTPUT.B32;1 Page 76
(24)

00000000G	00	00000000G	8F	DD	0002F	PUSHL	#DIFS_WRITEERR	:
	50		05	FB	00035	CALLS	#5, LTBSSIGNAL	:
			01	DD	0003C	MOVL	#1, R0	:
			04	0003F	18:	RET		: 2364

; Routine Size: 64 bytes, Routine Base: \$CODE\$ + 1133

DE

```
2087 2366 1 ROUTINE insert_linenum (rdb, linedesc, condense) =
2088 2367 BEGIN
2089 2368
2090 2369
2091 2370
2092 2371
2093 2372
2094 2373
2095 2374
2096 2375
2097 2376
2098 2377
2099 2378
2100 2379
2101 2380
2102 2381
2103 2382
2104 2383
2105 2384
2106 2385
2107 2386
2108 2387
2109 2388
2110 2389
2111 2390
2112 2391
2113 2392
2114 2393
2115 2394
2116 2395
2117 2396
2118 2397
2119 2398
2120 2399
2121 2400
2122 2401
2123 2402
2124 2403
2125 2404
2126 2405
2127 2406
2128 2407
2129 2408
2130 2409
2131 2410
2132 2411
2133 2412
2134 2413
2135 2414
2136 2415
2137 2416
2138 2417
2139 2418
2140 2419
2141 2420
2142 2421
2143 2422

ROUTINE insert_linenum (rdb, linedesc, condense) =
BEGIN
--
FUNCTIONAL DESCRIPTION:
    Insert the line number of the specified record in the output buffer.
INPUTS:
    rdb =      The address of the RDB of the record whose
                line number is to be inserted.
    linedesc = The address of a string descriptor for the string that
                the number is to be inserted at the end of.
    condense = A flag that is true if the inserted number should not
                be padded with more than one blank on each side.
                The default is to pad the number out to DIF$C_LINENUM
                spaces.
OUTPUTS:
    The line number is inserted in the output buffer, at the specified
    position.
ROUTINE VALUES:
    Always true
--
MAP
    rdb : REF BBLOCK,
    linedesc : REF BBLOCK;
LOCAL
    numdesc : BBLOCK [dsc$b_s_bln],
    numbuf : BBLOCK [dif$c_linenum];
    numdesc [dsc$b_class] = dsc$b_class_s;
    numdesc [dsc$b_length] = dif$c_linenum - 1;
    numdesc [dsc$a_pointer] = numbuf;
    OTSSCVT_L_TI (rdb [rdb$l_number], numdesc);
    IF .condense
    THEN BEGIN
        WHILE (CH$RCHAR (.numdesc [dsc$a_pointer]) EQL 'X '20')
        DO BEGIN
            numdesc [dsc$a_pointer] = .numdesc [dsc$a_pointer] + 1;
            numdesc [dsc$b_length] = .numdesc [dsc$b_length] - 1;
        END;
        CH$UCHAR (' ', .linedesc [dsc$a_pointer]
```



```
2144      linedesc [dsc$w_length] = .linedesc [dsc$w_length];  
2145      END;  
2146  
2147      CHSMOVE (.numdesc [dsc$w_length], .numdesc [dsc$a_pointer],  
2148              .linedesc [dsc$w_length] + .linedesc [dsc$a_pointer]);  
2149      CHSWCHAR ('C', .linedesc [dsc$a_pointer] + .linedesc [dsc$w_length]  
2150              + .numdesc [dsc$w_length]);  
2151      linedesc [dsc$w_length] = .linedesc [dsc$w_length]  
2152              + .numdesc [dsc$w_length] + 1;  
2153  
2154      RETURN true;  
2155      END;  
2156
```

		00FC 00000 INSERT_LINENUM:			
	SE	10	C2 00002	WORD	Save R2,R3,R4,R5,R6,R7
	0B AE	01	90 00005	SUBL2	#16, SP
	0B AE	05	B0 00009	MOVB	#1, NUMDESC+3
	0C AE	0E	9E 0000D	MOVW	#5, NUMDESC
		0B AE	9F 00011	MOVAB	NUMBUF, NUMDESC+4
7E	04 AC	04	C1 00014	PUSHAB	NUMDESC
00000000G	00	02	FB 00019	ADDL3	#4, RDB, -(SP)
	1E	0C AC	E9 00020	CALLS	#2, OTS\$CVT_L_T1
	20	0C BE	91 00024	BLBC	CONDENSE, 3\$
		0B	12 00028	CMPB	@NUMDESC+4, #32
		0C AE	D6 0002A	BNEQ	2\$
		0B AE	B7 0002D	INCL	NUMDESC+4
		F2	11 00030	DECW	NUMDESC
	50	0B AC	D0 00032	BRB	1\$
	51	04 A0	C0 00039	MOVL	LINEDESC, R0
	51	20	90 0003D	MOVZWL	(R0), R1
	61	60	3C 00036	ADDL2	4(R0), R1
		60	B6 00040	MOVB	#32, (R1)
	57	0B AE	3C 00042	INCL	(R0)
	56	0B AC	D0 00046	MOVZWL	NUMDESC, R7
	50	66	3C 0004A	MOVL	LINEDESC, R6
	50	04 A6	C0 0004D	MOVZWL	(R6), R0
60	0C BE	57	28 00051	ADDL2	4(R6), R0
	50	66	3C 00056	MOVC3	R7, @NUMDESC+4, (R0)
	50	04 A6	C0 00059	MOVZWL	(R6), R0
	6740	20	90 0005D	ADDL2	4(R6), R0
	50	66	3C 00061	MOVB	#32, (R7)[R0]
	51	01 A740	9E 00064	MOVZWL	(R6), R0
	66	51	B0 00069	MOVAB	1(R7)[R0], R1
	50	01	D0 0006C	MOVW	R1, (R6)
		04	0006F	MOVL	#1, R0
				RET	

; Routine Size: 112 bytes. Routine Base: \$CODE\$ + 1173

```
2158 2436 GLOBAL ROUTINE init_hex_octal =
2159 2437 BEGIN
2160 2438
2161 2439 --
2162 2440
2163 2441 FUNCTIONAL DESCRIPTION:
2164 2442
2165 2443     Prepare for hex or octal output by building the required FAO
2166 2444     descriptors and initializing some global variables.
2167 2445
2168 2446 INPUTS:
2169 2447
2170 2448     None
2171 2449
2172 2450 OUTPUTS:
2173 2451
2174 2452     None
2175 2453
2176 2454 ROUTINE VALUES:
2177 2455
2178 2456     Always true
2179 2457
2180 2458 --
2181 2459
2182 2460 LOCAL
2183 2461     fulldesc : BBLOCK [dsc$sc_s_bln],
2184 2462     offsetsize,
2185 2463     partdesc : BBLOCK [dsc$sc_s_bln];
2186 2464
2187 2465
2188 2466     Calculate the size of each entry in a line.
2189 2467
2190 2468 IF .dif$gl_flags [dif$sv_hex]
2191 2469 THEN offsetsize = 9
2192 2470 ELSE offsetsize = 12;
2193 2471
2194 2472
2195 2473     Calculate the number of longwords to output per line.
2196 2474     Force it to be a power of two.
2197 2475
2198 2476 dif$gl_entsperline = MAXU (
2199 2477     (.dif$gl_width - dif$sc_linenum - 1) / (.offsetsize + dif$sc_entrysize),
2200 2478     1);
2201 2479 IF (.dif$gl_entsperline AND (.dif$gl_entsperline-1)) NEQ 0
2202 2480 THEN
2203 2481     dif$gl_entsperline = .dif$gl_entsperline AND (.dif$gl_entsperline-1);
2204 2482
2205 2483
2206 2484     Calculate the line width.
2207 2485
2208 2486 dif$gl_dumpwidth = .dif$gl_entsperline * (.offsetsize + dif$sc_entrysize) + 8;
2209 2487
2210 2488
2211 2489     Pick the appropriate FAO descriptors to use.
2212 2490
2213 2491 dif$gl_faofulldesc [dsc$w_length] = dif$sc_maxfaosiz;
2214 2492 dif$gl_faofulldesc [dsc$a_pointer] = dif$gl_faofullbuf;
```

```
2215 2493 2 dif$gl_faopartdesc [dsc$w_length] = dif$c_maxfaosiz;
2216 2494 dif$gl_faopartdesc [dsc$a_pointer] = dif$gl_faopartbuf;
2217 2495
2218 2496 IF dif$gl_flags [dif$u_hex]
2219 2497 THEN BEGIN
2220 2498     fulldesc [dsc$w_length] = .hexfull [0];
2221 2499     fulldesc [dsc$a_pointer] = hexfull [1];
2222 2500     partdesc [dsc$w_length] = .hexpart [0];
2223 2501     partdesc [dsc$a_pointer] = hexpart [1];
2224 2502     END
2225 2503 ELSE BEGIN
2226 2504     fulldesc [dsc$w_length] = .octfull [0];
2227 2505     fulldesc [dsc$a_pointer] = octfull [1];
2228 2506     partdesc [dsc$w_length] = .octpart [0];
2229 2507     partdesc [dsc$a_pointer] = octpart [1];
2230 2508     END;
2231 2509
2232 2510 SYSSFAO (fulldesc, dif$gl_faofulldesc, dif$gl_faofulldesc,
2233 2511     .dif$gl_entspersline, .dif$gl_entspersline * dif$c_entsize);
2234 2512 SYSSFAO (partdesc, dif$gl_faopartdesc, dif$gl_faopartdesc,
2235 2513     .dif$gl_entspersline * dif$c_entsize);
2236 2514
2237 2515 RETURN true;
2238 2516 END;
```

		DIFC	00000	.ENTRY	INIT HEX OCTAL, Save R2,R3,R4,R5,R6,R7,R8		
58	00000000G	00	9E	00002	MOVAB	SYSSFAO, R8	2436
57	00000000G	00	9E	00009	MOVAB	DIF\$GL_FAOPARTDESC, R7	
56	00000000G	00	9E	00010	MOVAB	DIF\$GL_FAOFULLDESC, R6	
55	00000000G	00	9E	00017	MOVAB	DIF\$GL_ENTSPERLINE, R5	
54	00000000G	EF	9E	0001E	MOVAB	HEXFUL, R4	
53	00000000G 00	10	C2	00025	SUBL2	#16, SP	
01		01	EF	00028	EXTZV	#1, #1, DIF\$GL_FLAGS, R3	2468
05		53	E9	00031	BLBC	R3, 1\$	
50		09	D0	00034	MOVL	#9, OFFSETSIZE	2469
		03	11	00037	BRB	2\$	
50		0C	D0	00039	MOVL	#12, OFFSETSIZE	2470
51	00000000G	07	C3	0003C	SUBL3	#7, DIF\$GL_WIDTH, R1	2477
50		04	C0	00044	ADDL2	#4, R0	
51		50	C6	00047	DIVL2	R0, R1	
		03	12	0004A	BNEQ	3\$	2476
51		01	D0	0004C	MOVL	#1, R1	
65		51	D0	0004F	MOVL	R1, DIF\$GL_ENTSPERLINE	
52	FF	A1	9E	00052	MOVAB	-1(R1), R2	2479
52		51	D3	00056	BITL	R1, R2	
		06	13	00059	BEQL	4\$	
51		52	D2	0005B	MCOML	R2, R1	2481
65		51	CA	0005E	BICL2	R1, DIF\$GL_ENTSPERLINE	
52		65	D0	00061	MOVL	DIF\$GL_ENTSPERLINE, R2	2486
50		52	C4	00064	MULL2	R2, R0	
00000000G	00	A0	9E	00067	MOVAB	8(R0), DIF\$GL_DUMPWIDTH	
66		28	80	0006F	MOVW	#40, DIF\$GL_FAOFULLDESC	2491
04	A6 00000000G	00	9E	00072	MOVAB	DIF\$GL_FAOFULLBUF, DIF\$GL_FAOFULLDESC+4	2492

	04	67		28	80	0007A	MOVW	#40, DIF\$GL_FAOPARTDESC	2493
		A7	00000000G	00	9E	0007D	MOVAB	DIF\$GL_FAOPARTBUF, DIF\$GL_FAOPARTDESC+4	2494
		14		53	EB	00085	BLBC	R3, 58	2496
	08	AE		64	9B	00088	MOVZBW	HEXFULL, FULLDESC	2498
	0C	AE	01	A4	9E	0008C	MOVAB	HEXFULL+1, FULLDESC+4	2499
		6E	4C	A4	9B	00091	MOVZBW	HEXPART, PARTDESC	2500
	04	AE	4D	A4	9E	00095	MOVAB	HEXPART+1, PARTDESC+4	2501
				15	11	0009A	BRB	68	2496
	08	AE	6C	A4	9B	0009C	MOVZBW	OCTFULL, FULLDESC	2504
	0C	AE	6D	A4	9E	000A1	MOVAB	OCTFULL+1, FULLDESC+4	2505
		6E	00B8	C4	9B	000A6	MOVZBW	OCTPART, PARTDESC	2506
	04	AE	00B9	C4	9E	000AB	MOVAB	OCTPART+1, PARTDESC+4	2507
7E		52		02	78	000B1	ASHL	#2, R2, -(SP)	2511
				52	DD	000B5	PUSHL	R2	
				56	DD	000B7	PUSHL	R6	2510
				56	DD	000B9	PUSHL	R6	
			18	AE	9F	000BB	PUSHAB	FULLDESC	
		68		05	FB	000BE	CALLS	#5, SYSS\$FAO	
7E		65		02	78	000C1	ASHL	#2, DIF\$GL_ENTSPERLINE, -(SP)	2513
				57	DD	000C5	PUSHL	R7	2512
				57	DD	000C7	PUSHL	R7	
			0C	AE	9F	000C9	PUSHAB	PARTDESC	
		68		04	FB	000CC	CALLS	#4, SYSS\$FAO	
		50		01	DD	000CF	MOVL	#1, R0	2515
				04	00	000D2	RET		2516

; Routine Size: 211 bytes, Routine Base: \$CODE\$ + 11E3


```
2240 2517 1 ROUTINE get_rfa_text (fdb, linedesc, width) =
2241 2518 BEGIN
2242 2519
2243 2520 ++
2244 2521
2245 2522 FUNCTIONAL DESCRIPTION:
2246 2523
2247 2524     Get a record from an input file using its RFA, and insert it in
2248 2525     the output line.
2249 2526
2250 2527 INPUTS:
2251 2528
2252 2529     fdb =       The address of the FDB for the input file.
2253 2530                CURREC specifies the record to be read.
2254 2531
2255 2532     linedesc =  The address of a character string descriptor for the
2256 2533                output line.
2257 2534
2258 2535     width =     The size of the input buffer specified by linedesc.
2259 2536
2260 2537 OUTPUTS:
2261 2538
2262 2539     The read record is appended to linedesc.
2263 2540
2264 2541 ROUTINE VALUES:
2265 2542
2266 2543     Always true
2267 2544
2268 2545 --
2269 2546
2270 2547 MAP
2271 2548     fdb : REF BBLOCK,
2272 2549     linedesc : REF BBLOCK;
2273 2550
2274 2551 LOCAL
2275 2552     prefix_len,
2276 2553     rab : REF BBLOCK,
2277 2554     rdb : REF BBLOCK,
2278 2555     status;
2279 2556
2280 2557 rab = .fdb [fdb$l_rabptr];
2281 2558 rdb = .fdb [fdb$l_currec];
2282 2559
2283 2560 rab [rab$b_rac] = rab$c_rfa;
2284 2561 CHSMOVE (rfa$c_size, rdb [rdb$b_rfa], rab [rab$b_rfa]);
2285 2562
2286 2563 IF NOT (status = $GET (RAB = .rab))
2287 2564 THEN SIGNAL_STOP (dif$_readerr, 1, .fdb [fdb$l_fildesc],
2288 2565                  .status, .rab [rab$l_stv]);
2289 2566
2290 2567 IF .rab [rab$b_rsz] GTR 0
2291 2568 THEN BEGIN
2292 2569     prefix_len = .linedesc [dsc$b_length];
2293 2570     linedesc [dsc$b_length] = MIN0 (.width,
2294 2571                                     linedesc [dsc$b_length] + .rab [rab$b_rsz]);
2295 2572     CHSMOVE (.linedesc [dsc$b_length] - .prefix_len,
2296 2573             .rab [rab$l_rbf], .linedesc [dsc$a_pointer] + .prefix_len);
```

```
2297      END;
2298
2299
2300
2301      After getting a record by RFA, we must reset the RMS pointers for
2302      subsequent sequential reads to work.
2303
2304      rdb = fdb [fdb$l_lastrec];
2305      IF rdb [rdb$w_eot]
2306      THEN rdb = fdb [fdb$l_lastrfa];
2307      IF rdb NEQ 0
2308      THEN BEGIN
2309          CH$MOVE (rfa$c_size, rdb [rdb$w_rfa], rab [rab$w_rfa]);
2310          IF NOT (status = $FIND (RAB = .rab))
2311          THEN SIGNAL_STOP (dif$readerr, 1, fdb [fdb$l_fildesc], ! If error, then signal
2312                          .status, .rab [rab$l_stv]); ! Reset RAB
2313          rab [rab$b_rac] = rab$c_seq;
2314          IF NOT (status = $GET (RAB = .rab))
2315          THEN SIGNAL_STOP (dif$readerr, 1, fdb [fdb$l_fildesc], ! If error, then signal
2316                          .status, .rab [rab$l_stv]);
2317      END;
2318
2319      RETURN true;
2320      END;
```

.EXTRN SYS\$GET, SYS\$FIND

OFFC 00000 GET_RFA_TEXT:						
5B	00000000G	00	9E 00002	.WORD	Save R2,R3,R4,R5,R6,R7,R8,R9,R10,R11	2517
5A	00000000G	8F	D0 00009	MOVAB	LIB\$STOP, R11	
57	04	AC	D0 00010	MOVL	#DIF\$ READERR, R10	
56	30	A7	D0 00014	MOVL	FDB, R7	2557
58		67	D0 00018	MOVL	48(R7), RAB	
1E	A6	02	90 0001B	MOVL	(R7), RDB	2558
OC	A8	06	28 0001F	MOVB	#2, 30(RAB)	2560
		56	DD 00025	MOVC3	#6, 12(RDB), 16(RAB)	2561
00000000G	00	01	FB 00027	PUSHL	RAB	2563
59	OF	50	D0 0002E	CALLS	#1, SYS\$GET	
		59	E8 00031	MOVL	R0, STATUS	
	OC	A6	DD 00034	BLBS	STATUS, 1\$	
		59	DD 00037	PUSHL	12(RAB)	2565
	38	A7	DD 00039	PUSHL	STATUS	
		01	DD 0003C	PUSHL	56(R7)	2564
		5A	DD 0003E	PUSHL	#1	
	6B	05	FB 00040	PUSHL	R10	
		22	A6 B5 00043 1\$:	CALLS	#5, LIB\$STOP	
		2D	13 00046	TSTW	34(RAB)	2567
	50	08	AC D0 00048	BEQL	3\$	
	51	60	3C 0004C	MOVL	LINEDESC, R0	2569
	53	60	3C 0004F	MOVZWL	(R0), PREFIX_LEN	
	52	22	A6 3C 00052	MOVZWL	(R0), R3	2571
	53	52	C0 00056	MOVZWL	34(RAB), R2	
	52	OC	AC D0 00059	ADDL2	R2, R3	
	53	52	D1 0005D	MOVL	WIDTH, R2	
		03	1B 00060	CMPL	R2, R3	
				BLEQU	2\$	

04 B041	28	B6	08	53	D0 00062	MOVL	R3, R2	:	2570
		53		52	B0 00065	MOVW	R2, (R0)	:	2572
		53		60	3C 00068	MOVZWL	(R0), R3	:	2573
		53		51	C2 0006B	SUBL2	PREFIX_LEN, R3	:	2580
		58		53	28 0006E	MOVCL	R3, @40(RAB), @4(R0)[PREFIX_LEN]	:	2581
04	04	08	A8	02	E1 00079	BBC	#2, 8(RDB), 4\$:	2582
		58	1C	A7	D0 0007E	MOVL	28(R7), RDB	:	2583
10	A6	0C	A8	45	13 00082	BEQL	6\$:	2585
		00		06	28 00084	MOVCL	#6, 12(RDB), 16(RAB)	:	2586
	00000000G	00		56	DD 0008A	PUSHL	RAB	:	2588
		59		01	FB 0008C	CALLS	#1, SYSS\$FIND	:	2587
		0F		50	D0 00093	MOVL	R0, STATUS	:	2589
			0C	59	E8 00096	BLBS	STATUS, 5\$:	2590
				A6	DD 00099	PUSHL	12(RAB)	:	2592
			38	55	DD 0009C	PUSHL	STATUS	:	2591
				A7	DD 0009E	PUSHL	56(R7)	:	2595
				01	DD 000A1	PUSHL	#1	:	2596
				5A	DD 000A3	PUSHL	R10	:	
		6B		05	FB 000A5	CALLS	#5, LIB\$STOP	:	
			1E	A6	94 000A8	CLRB	30(RAB)	:	
		00		56	DD 000AB	PUSHL	RAB	:	
	00000000G	59		01	FB 000AD	CALLS	#1, SYSS\$GET	:	
		0F		50	D0 000B4	MOVL	R0, STATUS	:	
			0C	59	E8 000B7	BLBS	STATUS, 6\$:	
				A6	DD 000BA	PUSHL	12(RAB)	:	
			38	59	DD 000BD	PUSHL	STATUS	:	
				A7	DD 000BF	PUSHL	56(R7)	:	
				01	DD 000C2	PUSHL	#1	:	
				5A	DD 000C4	PUSHL	R10	:	
		6B		05	FB 000C6	CALLS	#5, LIB\$STOP	:	
		50		01	D0 000C9	MOVL	#1, R0	:	
				04	000CC	RET		:	

; Routine Size: 205 bytes, Routine Base: \$CODE\$ + 12B6

: 2320 2597 1
: 2321 2598 1 END
: 2322 2599 0 ELUDOM

! Of module

.EXTRN LIB\$SIGNAL, LIB\$STOP

PSECT SUMMARY

Name	Bytes	Attributes
\$OWNS	401	NOVEC, WRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$CODE\$	4995	NOVEC, NOWRT, RD, EXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)
\$SPLITS	562	NOVEC, NOWRT, RD, NOEXE, NOSHR, LCL, REL, CON, NOPIC, ALIGN(2)

DIF OUTPUT
V04=000

D 11
13-Sep-1984 23:43:35
14-Sep-1984 12:19:23

VAX-11 Bliss-32 V4.0-742
DISK\$VMSMASTER:[DIF.SRC]OUTPUT.B32;1 Page 85
(27)

Library Statistics

File	----- Total	Symbols Loaded	----- Percent	Pages Mapped	Processing Time
_S255\$DUA28:[SYSLIB]STARLET.L32;1	9776	24	0	581	00:01.0

COMMAND QUALIFIERS

BLISS/CHECK=(FIELD,INITIAL,OPTIMIZE)/LIS=LIS\$:OUTPUT/OBJ=OBJ\$:OUTPUT MSRC\$:OUTPUT/UPDATE=(ENHS:OUTPUT)

: Size: 4995 code + 963 data bytes
: Run Time: 01:21.8
: Elapsed Time: 02:46.0
: Lines/CPU Min: 1906
: Lexemes/CPU-Min: 20144
: Memory Used: 262 pages
: Compilation Complete

0103 AH-BT13A-SE
VAX/VMS V4.0

DIGITAL EQUIPMENT CORPORATION
CONFIDENTIAL AND PROPRIETARY

DIFMSG
LIS

MAIN
LIS

DIR

DIRECTORY
MAP

DIRECTORY
LIS

DIRECTDEF
REQ

OUTPUT
LIS

DISPDEF
SDI

DIRECTMSG
LIS